CENTER FOR CONNECTED AND AUTOMATED TRANSPORTATION

Report No. 13 **December 2021**

Project Start Date: 2/1/2019

Project End Date: 12/31/2021

Real-time Distributed Optimization of Traffic Signal Timing

Xinyu Fei[2]
Xian Yu[2]
Xingmin Wang[1]
Tian Mi[1]
Yafeng Yin[1,2] (PI)
Siqian Shen[2] (Co-PI)
Yiheng Feng[3] (Co-PI)

[1]Department of Civil and Environmental Engineering, University of Michigan
[2]Department of Industrial and Operations Engineering, University of Michigan
[3]University of Michigan Transportation Research Institute

UMTRI | The University of Akron | Illinois Center for Transportation | Purdue University | Washtenaw Community College | Central State University

CENTER FOR CONNECTED AND AUTOMATED TRANSPORTATION

DISCLAIMER

Funding for this research was provided by the Center for Connected and Automated Transportation under Grant No. 69A3551747105 of the U.S. Department of Transportation, Office of the Assistant Secretary for Research and Technology (OST-R), University Transportation Centers Program. The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the Department of Transportation, University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

Suggested APA Format Citation: Yin, Yafeng, Shen, Siqian, Feng, Yiheng, Fei, Xinyu, Yu, Xian, Wang Xingmin, & Mi, Tian, (2021) "Real-time Distributed Optimization of Traffic Signal Timing". CCAT Project No. 13, Center for Connected and Automated Transportation, University of Michigan.
DOI: 10.7302/6942

Contacts

For more information:

PI Name: Yafeng Yin
Address: 2120 GG Brown, Ann Arbor, Michigan
Phone: (734) 764-8249
Email: yafeng@umich.edu

Co-PI Name : Siqian Shen
Address: 2793 IOE, 1205 Beal Avenue, Ann Arbor, Michigan
Email: siqian@umich.edu

Co-PI Name: Yiheng Feng
Address: 550 Stadium Mall Drive West Lafayette, Indiana
Phone: (765) 496-5025
Email: feng333@purdue.edu

CCAT
University of Michigan Transportation Research Institute
2901 Baxter Road
Ann Arbor, MI 48152
umtri-ccat@umich.edu
(734) 763-2498

CENTER FOR CONNECTED AND AUTOMATED TRANSPORTATION

Technical Report Documentation Page

1. Report No. CCAT Report No. 13	**2. Government Accession No.** Leave blank – not used	**3. Recipient's Catalog No.** Leave blank - not used
4. Title and Subtitle Real-time Distributed Optimization of Traffic Signal Timing DOI: 10.7302/6942		**5. Report Date** December 31, 2021 **6. Performing Organization Code** Enter any/all unique numbers assigned to the performing organization, if applicable.
7. Author(s) Xinyu Fei, https://orcid.org/0000-0001-7010-8664 Xian Yu, https://orcid.org/0000-0003-1059-5303 Xingmin Wang, https://orcid.org/0000-0003-0435-2786 Tian Mi, https://orcid.org/0000-0002-3780-3216 Yafeng Yin, https://orcid.org/0000-0003-3117-5463 Siqian Shen, https://orcid.org/0000-0002-2854-163X Yiheng Feng, https://orcid.org/0000-0001-5656-3222		**8. Performing Organization Report No.** Enter any/all unique alphanumeric report numbers assigned by the performing organization, if applicable.
9. Performing Organization Name and Address Center for Connected and Automated Transportation University of Michigan Transportation Research Institute 2901 Baxter Road Ann Arbor, MI 48109 University of Michigan 2350 Hayward Ann Arbor, MI 48109		**10. Work Unit No.** **11. Contract or Grant No.** Contract No. 69A3551747105
12. Sponsoring Agency Name and Address U.S. Department of Transportation Office of the Assistant Secretary for Research and Technology 1200 New Jersey Avenue, SE Washington, DC 20590		**13. Type of Report and Period Covered** February 2019 – December 2021 **14. Sponsoring Agency Code** OST-R
15. Supplementary Notes Conducted under the U.S. DOT Office of the Assistant Secretary for Research and Technology's (OST-R) University Transportation Centers (UTC) program.		

UMTRI · The University of Akron · Illinois Center for Transportation · Purdue University · Washtenaw Community College · Central State University

CENTER FOR CONNECTED AND AUTOMATED TRANSPORTATION

16. Abstract

Leveraging recent advancements in distributed optimization and reinforcement learning, and the growing connectivity and computational capability of vehicles and infrastructure, we propose to advance real-time adaptive signal control via distributed control and optimization. This report consists of three parts. Part 1 develops distributed algorithms for solving a traffic signal timing optimization problem, which is formulated as a mixed-integer programming model. Specifically, the alternating direction method of multipliers (ADMM) is employed, and a two-stage stochastic cell transmission model (CTM) that considers the uncertainty of traffic demand and vehicle turning ratios is considered. Part 2 proposes a framework that utilizes reinforcement learning to optimize a max pressure controller considering the phase switching loss. The max pressure control is modified by introducing a switching curve, and the proposed control method is proved throughput-optimal in a store-and-forward network. Then the theoretical control policy is extended by using a distributed approximation and position-weighted pressure so that the policy-gradient reinforcement learning algorithms can be utilized to optimize the parameters in the policy network including the switching curve and the weigh curve. Part 3 applies reinforcement learning to traffic signal control in a multi-agent scheme, considering the data availability and implementability. The information extracted from traffic cameras is used to define the state of the agents; the action design is aligned with the NEMA dual-ring convention and bounded by a safety constraint, and the coordination is achieved by a shared reward structure among agents.

17. Key Words	**18. Distribution Statement**
Traffic signal control, distributed optimization, max pressure control, reinforcement learning	No restrictions.

19. Security Classif. (of this report)	**20. Security Classif. (of this page)**	**21. No. of Pages**	**22. Price**
Unclassified	Unclassified	75	Leave blank – not used

Form DOT F 1700.7 (8-72)

Reproduction of completed page authorized

UMTRI | The University of Akron | Illinois Center for Transportation | Purdue University | Washtenaw Community College | Central State University

Contents

Abstract

Leveraging recent advancements in distributed optimization and reinforcement learning, and the growing connectivity and computational capability of vehicles and infrastructure, we propose to advance real-time adaptive signal control via distributed control and optimization. This report consists of three parts. Part 1 develops distributed algorithms for solving a traffic signal timing optimization problem, which is formulated as a mixed-integer programming model. Specifically, the alternating direction method of multipliers (ADMM) is employed, and a two-stage stochastic cell transmission model (CTM) that considers the uncertainty of traffic demand and vehicle turning ratios is considered. Part 2 proposes a framework that utilizes reinforcement learning to optimize a max pressure controller considering the phase switching loss. The max pressure control is modified by introducing a switching curve, and the proposed control method is proved throughput-optimal in a store-and-forward network. Then the theoretical control policy is extended by using a distributed approximation and position-weighted pressure so that the policy-gradient reinforcement learning algorithms can be utilized to optimize the parameters in the policy network including the switching curve and the weight curve. Part 3 applies reinforcement learning to traffic signal control in a multi-agent scheme, considering the data availability and implementability. The information extracted from traffic cameras is used to define the state of the agents; the action design is aligned with the NEMA dual-ring convention and bounded by a safety constraint, and the coordination is achieved by a shared reward structure among agents.

Chapter 1

Introduction

Traffic congestion is a pressing issue globally and can be mitigated via effective traffic signal control. The overarching goal of the research project is to revolutionize real-time adaptive signal control via distributed optimization. Optimal timing of traffic signals across a traffic network has been extensively studied in the literature. The optimization models formulated typically contain discrete variables and nonconvex objectives or constraints, and solving them in a centralized fashion has been proved to be infeasible for real-time large-scale implementation. Consequently, the state-of-the-practice is to either coordinate signals at the corridor level via actuated control logic or adopt a hierarchical (decentralized) approach to decompose the original problem (network) into smaller problems (regions) for better computation efficiency. None of these implementations achieves global or near-global optimality. We believe it is time to overcome this challenge that has puzzled traffic engineers for more than half a century. Our belief is based on the following three observations: (i) multi-core processors have become ubiquitous. It is thus natural to look to parallelism as a mechanism for solving large-scale optimization tasks. Recent developments in distributed optimization have demonstrated its great potential to be computationally superior to centralized algorithms in terms of efficiency and scale (Boyd et al. (2004); Timotheou et al. (2014)); (ii) vehicles and transportation infrastructure are becoming "smarter" and connected. The new generation of automobiles and roadside devices are equipped with powerful on-board computing hardware and are capable of communicating with each other as well as performing complex computational tasks. The deployment of these vehicles and roadside devices provides opportunities to distribute the computation task to the vehicle and intersection level, in the same spirit as the notion of edge computing (Garcia Lopez et al. (2015)); (iii) recent advances in blockchain technology have made it an enabler for implementing distributed and secured traffic signal control. Blockchains allow us to establish a distributed peer-to-peer (i.e., vehicle-to-vehicle) network where non-trusting members can interact with each other without a trusted intermediary in a verifiable manner. Smart contacts mechanism enable connected vehicles to act as fully independent economic agents that negotiate the right-of-way through intersections (Yuan and Wang (2016)).

This report documents our preliminary attempts on leveraging these technology advances to achieve real-time large-scale distributed control of traffic signals. The report is organized as follows. Chapter 2 proposes distributed optimization algorithms to design network-based traffic signal control under uncertain traffic demand and vehicle turning ratio. Chapter 3 proposes a framework that utilizes reinforcement learning algorithms to optimize a max pressure controller considering the phase switching loss. Chapter 4 uses pure reinforcement learning method to examine the traffic signal control problem in a multi-agent framework.

Chapter 2

Optimization and Decentralized Algorithms for Traffic Signal Control

2.1 Introduction

Traffic congestion can lead to significant travel delay, and occurs when the traffic volume exceeds road capacity (Isa et al., 2014). Over the past few decades, in many countries, city scales have increased significantly and more populations have migrated from countryside to urban areas. As a result, the number of vehicles and the total traveled miles increase significantly, resulting in rapidly growing congestion issues in cities of all sizes worldwide. According to Schrank et al. (2019), in 2017, traffic congestion caused urban Americans to travel extra 8.8 billion hours and to purchase extra 3.3 billion gallons of fuel. Taking the city of Detroit in the United States as an example, in 2019, each driver lost 39 hours on the road on average due to traffic delay (Levin, 2020).

Various approaches have been proposed for addressing traffic congestion issues on road networks, including congestion pricing, road expansion, traffic routing, and traffic signal control. Among them, traffic signal control can effectively mitigate congestion by changing the time length of traffic signals at road intersections (Isa et al., 2014). At present, in the United States, most traffic signals operate time-dependent signal plans determined by a central operator, requiring periodic manual updates due to time-varying traffic demand. The increase in traffic congestion raises the need for more efficient and effective traffic signal control plans.

In this paper, we formulate mixed-integer programming models and develop distributed algorithms for solving network-based traffic signal control problems, to maximize vehicle throughput on traffic networks. Different from most of the existing literature, we also take into account traffic demand uncertainty and stochastic vehicle turning ratios. Our goal is to establish an optimization-based control paradigm to speed up solutions to large-scale traffic signal control *in a distributed manner*. We demonstrate the results by testing diverse instances, generated using synthetic data and also real-world road networks and traffic data. Via out-of-sample tests of solutions given by different methods in a variety of scenarios, we show that the signal control plans produced by our methods perform consistently better than plans solved using off-the-shelf optimization solvers, or models that do not consider uncertainty. Furthermore, our decomposition and decentralized algorithms can significantly reduce the computational time and produce reliable traffic signal plans depending on distributions of uncertain traffic throughout a day for moderate-sized city networks.

The remainder of this paper is organized as follows. In Section 2.2, we review the most relevant literature on traffic signal control and optimization methods we use in this paper. In Section 2.3, we first formulate the deterministic traffic signal control problems as a mixed-integer program (MIP). Then, we take uncertain demand and turning ratios into account and extend the deterministic model to a two-stage stochastic integer program. In Section 2.4, we develop spatially distributed algorithms for solving the stochastic optimization model in a decentralized manner. In Section 2.5, we conduct numerical studies by testing a set of diverse instances. In Section 2.6, we conclude the paper and state

future research directions.

2.2 Literature review

We firstly review the literature about traffic flow modeling in Section 2.2.1. In Section 2.2.2, we review the most relevant work on distributed traffic signal control, especially the ones using ADMM for distributed convex optimization. In Section 2.2.3, we review the literature on traffic signal control under uncertainties.

2.2.1 Traffic Flow Modeling

The traffic flow models can be mainly classified as microscopic and macroscopic (see, e.g., Li and Ioannou, 2004; Hoogendoorn and Bovy, 2001). A microscopic model simulates the behavior of an individual vehicle by using dynamic variables to describe states such as location and velocity of vehicles (see e.g., Pipes, 1953; Wu and Brilon, 1999). On the other hand, a macroscopic model simulates the traffic situation using variables to describe overall states of all vehicles, including traffic density, volume, and average speed (Hoogendoorn and Bovy, 2001). Daganzo (1992) first develops a cell transmission model (CTM) to model traffic flows on highways. As a macroscopic model, the CTM divides roads into homogeneous sections and considers states of each section across discrete time steps. We refer the interested readers to Adacher and Tiriolo (2018) for a comprehensive review of different CTM variants and the appropriate environments for applying them. Compared to microscopic models, the computational cost of macroscopic models is significantly lower because of the aggregated traffic states and the resulting fewer number of variables and dynamic equations. Among them, the CTM can describe traffic flows more precisely, with an acceptable computational complexity and therefore, in this paper, our optimization framework for the traffic signal control is based on the CTM.

2.2.2 Distributed Traffic Signal Control

Lo (1999) builds an MIP based on the CTM to describe the traffic signal control problem but does not consider vehicle turning nor system uncertainty. The model is only tested on corridors instead of general road networks, mainly due to the exponentially increased number of variables and constraints in the latter case. With significantly increased urban network sizes, centralized MIP models become more difficult to solve via a central controller and gathering of global information in the whole traffic network. On the other hand, each intersection is physically isolated, making it natural to consider decentralized traffic signal control. Advancements in parallel computing and information communication technology allow efficient implementations of decentralized methods at individual intersections or clusters of a few intersections in a large road network, leading to more research focusing on how to design parallel and distributed traffic signal control schemes. In a representative work, Al Islam and Hajbabaie (2017) present a distributed-coordinated approach for signal timing optimization in connected urban street networks. The authors formulate the optimization problem of each intersection as an MIP based on the CTM and solve each model separately. Then, the intersections communicate to each other to obtain estimated traffic flows. However, the information is only used in the next time period and the signal plan of the current time period is not updated. Hence, the signal plan obtained by solving the problem for each intersection is sub-optimal.

Alternating Direction Method of Multipliers (ADMM) is a widely used optimization-based distributed algorithm. It was initially developed to solve convex programs that can be decomposed into multiple sub-clusters and the objective function is the summation of functions related to each subcluster. For convex programs, it is proved that both the objective value and solutions obtained from the ADMM converge to the optimum (Boyd et al., 2011). Timotheou et al. (2014) propose a deterministic MIP based on the CTM and solve them using a distributed algorithm that decomposes a traffic network into individual intersections. The authors apply ADMM to obtain the coordination between intersections as well as solving the problem for each intersection individually. However, vehicle turn-

ing is not taken into consideration. Moreover, the paper does not consider stochastic traffic demand or random vehicle turning.

2.2.3 Traffic Signal Control under Uncertainties

Most traffic signal control systems in the United States use time-of-day control, which applies different fixed-time signal timing plans during a day (Tang et al., 2019). In practice, traffic conditions in road networks can vary even for the same period over different days. Moreover, the movements of vehicles, described by turning ratios in our traffic signal control problem, are often uncertain (Yu and Recker, 2006). However, changing a signal timing plan in traffic networks is infrequent because it requires a complex series of work such as resetting controllers, tuning the parameters, and monitoring the traffic systems (Zhang et al., 2010). Therefore, a signal timing plan taking into account the uncertainties of traffic states is valuable. The main research in this area mainly takes a risk-neutral perspective or a robust view. Among them, Zhang et al. (2010) design a robust fixed-time signal timing plan for a corridor under the traffic demand uncertainty with known distribution based on the CTM. They ensure that the probability of the travel delay exceeding a given threshold is sufficiently small and the resulting model is solved by a heuristic approach.

In this paper, we aim to obtain a traffic signal plan with the best average performance using stochastic optimization. An important formulation of stochastic optimization models is the two-stage stochastic program (Shapiro, 1993) using the Sample Average Approximation (SAA) method (Shapiro and Homem-de Mello, 2000). Tong et al. (2015) propose a two-stage stochastic optimization model for an isolated intersection to minimize the mean of the traffic delay under the uncertainties of traffic demand. However, their model relaxes the integrality constraint and the authors neither extend to a general traffic network nor consider random vehicle turning. In our paper, we propose a two-stage stochastic MIP under the uncertainties of traffic demand and turning ratios based on SAA. Our distributed algorithms are combined with Benders decomposition for solving two-stage stochastic MIPs (Shapiro et al., 2014).

The main contributions of this paper are threefold. First, we consider a fixed-time traffic signal control problem and formulate it on general grid networks as a large-scale MIP based on the CTM, in which we not only consider movements along corridors but also turning movements of vehicles. Second, we extend the deterministic model to a stochastic counterpart by incorporating various types of input parameter uncertainties, such as random traffic demand and turning ratios. Third, we speed up the computation of the resultant problem in a decentralized way. Specifically, we firstly ignore the traffic flow relationship between intersections and obtain a signal timing plan for each intersection. Then, we coordinate the information of neighboring intersections and update the parameters of each intersection. Such a distributed computational framework can best utilize the advanced technologies in smart transportation such as distributed micro-computers – they can be installed at traffic intersections to use sensing information and wireless communication. In Table 2.1, we compare our work with the most relevant papers that use CTM, in terms of modeling method, instance scale, decision type, parameter assumption and solution approach.

Table 2.1: Comparison between our study and traffic signal control papers using CTM

Paper	Model	Scale	Decision	Parameter	Approach
Our work	MIP and two-stage SP* based on the CTM	network	fixed-time	stochastic traffic demand and turning ratio	decentralized
Lo (1999)	MIP based on the CTM	corridor	real-time	deterministic	centralized
Zhang et al. (2010)	robust MIP based on the CTM	corridor	fixed-time	stochastic traffic demand	centralized
Tong et al. (2015)	Two-stage SP*	isolated intersection	fixed-time	stochastic traffic demand	centralized
Timotheou et al. (2014)	MIP based on the CTM	network	real-time	deterministic	decentralized
Al Islam and Hajbabaie (2017)	MIP based on the CTM	network	real-time	deterministic	decentralized

*SP: Stochastic program;

2.3 Optimization Models for Traffic Signal Control

We first define the notation and introduce the CTM in Section 2.3.1. We formulate the fixed-time traffic signal control as a deterministic MIP based on the CTM in Section 2.3.2, in which we aim to maximize the total throughput (i.e., the number of vehicles going through the whole traffic network). Then in Section 2.3.3, we extend the deterministic optimization model to a two-stage stochastic optimization model using finite samples of the uncertain traffic demand and turning ratios. The model takes the initial traffic state, distributions of traffic demand and vehicle turning ratios as input parameters and output a signal control plan for traffic lights at all intersections.

2.3.1 Problem Description and Notation

Structure of the CTM Consider a road network $G(V, E)$ where V is the set of nodes and E is the set of arcs. Here, the nodes are signalized intersections and the arcs refer to road segments connecting pairs of the intersections. In this paper, we focus on road networks in which at most four arcs are connected to the same node, representing North, South, East and West directions and therefore the underlying network is a grid. Each arc is partitioned into homogeneous sections called cells, where the length of each cell is the distance traveled by a vehicle at normal speed without traffic congestion in a unit of time. Let $\mathcal{C} = \mathcal{E} \cup \mathcal{O} \cup \mathcal{D} \cup \mathcal{I} \cup \mathcal{M} \cup \mathcal{V}$ be the set of cells where $\mathcal{E}$ is the set of ordinary cells, $\mathcal{O}$ is the set of origin cells, $\mathcal{D}$ is the set of destination cells, $\mathcal{I}$ is the set of intersection cells, $\mathcal{M}$ is the set of merge cells and $\mathcal{V}$ is the set of diverge cells. Ordinary cells have both inflow and outflow of vehicles from other cells. Origin cells receive exogenous inflow and destination cells send outflow traffic outside the network. Intersection cells are cells where vehicles can choose to turn left, go straight or turn right. Merge cells receive inflow traffic from more than one cell, while diverge cells send outflow traffic to more than one cell. Let $d(c)$ be the set of cells sending their outflow to a cell $c \in \mathcal{C}$ and $p(c)$ be the set of cells receiving their inflow from a cell $c \in \mathcal{C}$. A signal phase of a signalized intersection is a set of all intersection cells that can be allowed to pass the intersection at the same time. Let $\mathcal{F}$ be the set of indices of all the phases of a signalized intersection. In our problem, the set of phases $\mathcal{F}$ is defined as $\{1, 2, 3, 4\}$, where $j = 1$ means turning left in East-West direction, $j = 2$ means going straight or turning right in East-West direction, $j = 3$ means turning left in North-South direction, and $j = 4$ means going straight or turning right in North-South direction. Let $\mathcal{R} = \{1, 2, \cdots, N_I\}$ be the set of intersections where N_I is the total number of intersections. Let I_{ij} be the set of the intersection cells of intersection i and phase j for all $i \in \mathcal{R}$ and $j \in \mathcal{F}$. We divide the whole time horizon into T time steps, represented by $\{1, \cdots, T\}$. Figure 2.1 depicts the way of dividing one intersection and its related arcs into cells in CTM. The arcs are divided into squares with different patterns, referring to different types of the aforementioned cells in a CTM.

Input Parameters For each cell $c \in \mathcal{C}$, let n_c^{init} be the number of initial vehicles inside a cell c. For each cell $c \in \mathcal{V}$ and each cell $c' \in d(c)$, let $\beta_{cc'}$ be the turning ratio of a diverging cell c moving towards the direction of an intersection cell c'. Notice that $\sum_{c' \in d(c)} \beta_{cc'} = 1$. For each cell $c \in \mathcal{C}$ and time step $t \in \{1, \cdots, T\}$, let Q_{ct} and N_{ct} represent the maximum number of vehicles that can flow through and reside into a cell c during time $[t, t+1)$. Let W_{ct} be the ratio between the shock-wave propagation speed and the flow-free speed of a cell c during time $[t, t+1)$. For each cell $c \in \mathcal{O}$ and each time step $t \in \{1, \cdots, T\}$, let D_{ct} be the number of vehicles entering an origin cell c during time $[t, t+1)$, defined as the traffic demand. We will use $n^{\text{init}}, \beta, Q, N, W, D$ to represent the vector forms of all the parameters introduced above, respectively. Moreover, let $G_{\min}$ and $G_{\max}$ be the minimum and maximum green time, indicating that any green signal set at time t cannot change during time $[t, t + G_{\min})$ and the green signal of the same direction cannot last for more than $G_{\max}$ time (Zhang and Wang, 2010), respectively. Let N_{cy} be the maximum number of cycles in the whole time horizon where traffic signal plans are the same in each cycle. In addition, we introduce a sufficiently large parameter U, a sufficiently small parameter ϵ and a weight parameter α to formulate constraints and the objective function in our models described later.

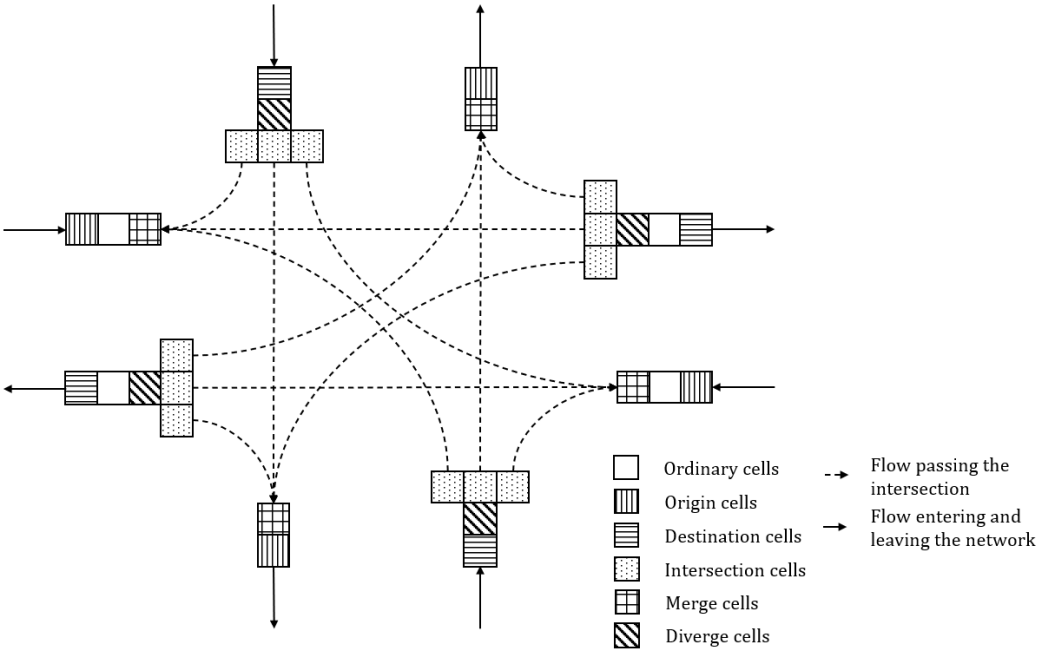


Figure 2.1: Example of a network with one intersection in the CTM. Bland squares are ordinary cells, squares with vertical stripes are origin cells, squares with horizontal stripes are destination cells, dotted squares are intersection cells, squares with grids are merge cells and squares with diagonal stripes are diverge cells. Dashed arrows indicate the network flow passing the intersection and solid arrows indicate the flow entering and leaving the whole traffic network.

Decision Variables For each cell $c \in \mathcal{C}$ and each time $t \in \{1, \cdots, T\}$, we define continuous variables y_{ct}, n_{ct} as the number of vehicles leaving cell c and inside cell c during time $[t, t+1)$, respectively. In our model, we describe the traffic signal control plan by determining the beginning time and ending time when the traffic signal is green for each phase and each cycle. We use offsets for intersections to describe the temporal difference between the start of the cycles of different intersections. For each intersection $i \in \mathcal{R}$, define continuous variables l_i to represent the cycle length and o_i to represent the offset of an intersection i. For each intersection $i \in \mathcal{R}$ and each phase $j \in \mathcal{F}$, continuous variables g_{ij} indicate the length of the time interval when the traffic signal of an intersection i and a phase j is green. For each intersection $i \in \mathcal{R}$, each phase $j \in \mathcal{F}$ and each cycle $j' \in \{1, \cdots, N_{\text{cy}}\}$, continuous variables $b_{ijj'}$ and $e_{ijj'}$ indicate the beginning and ending green time at intersection i, phase j during cycle j'. For each intersection $i \in \mathcal{R}$, phase $j \in \mathcal{F}$, cycle $j' \in \{1, \cdots, N_{\text{cy}}\}$ and time $t \in \{1, \cdots, T\}$, we define binary variables $z_{1ijj't}$ and $z_{2ijj't}$ to describe the relationship between the time step t, the beginning green time $b_{ijj'}$ and the ending green time $e_{ijj'}$ of the intersection i, the phase j and the cycle j', detailed in constraints (2.1n)–(2.1p) later. We use $y, n, w, l, o, b, e, g, z_1, z_2$ to represent the vector forms of all the variables mentioned above, respectively.

2.3.2 Deterministic Optimization Model

We formulate the deterministic traffic signal control problem as an MIP based on CTM below.

$$\min \quad -\sum_{c\in\mathcal{D}}\sum_{t=1}^{T} n_{ct} - \alpha \sum_{c\in\mathcal{C}}\sum_{t=1}^{T}(T-t)y_{ct} \tag{2.1a}$$

$$\text{s.t.} \quad y_{ct} \le n_{ct},\ \forall c \in \mathcal{C},\ t = 1, \cdots, T \tag{2.1b}$$

$$y_{ct} \le Q_{ct},\ \forall c \in \mathcal{E} \cup \mathcal{O} \cup \mathcal{M} \cup \mathcal{V},\ t = 1, \cdots, T \tag{2.1c}$$

$$y_{ct} \leq \sum_{j'=1}^{N_{\text{cy}}} (z_{1ijj't} + z_{2ijj't} - 1) Q_{ct},\ \forall c \in \mathcal{I}_{ij},\ \forall i \in \mathcal{R},\ \forall j \in \mathcal{F},\ t = 1, \cdots, T \tag{2.1d}$$

$$y_{ct} \leq Q_{c't},\ \forall c \in \mathcal{C}/\mathcal{V},\ \forall c' \in d(c),\ t = 1, \cdots, T \tag{2.1e}$$

$$\beta_{cc'} y_{ct} \leq Q_{c't},\ \forall c \in \mathcal{V},\ \forall i \in \mathcal{R},\ \forall j \in \mathcal{F},\ \forall c' \in d(c) k I_{ij},\ t = 1, \cdots, T \tag{2.1f}$$

$$y_{ct} \leq W_{c't}(N_{c't} - n_{c't}),\ \forall c \in \mathcal{C}/\mathcal{V},\ \forall c' \in d(c),\ t = 1, \cdots, T \tag{2.1g}$$

$$\beta_{cc'} y_{ct} \leq W_{c't}(N_{c't} - n_{c't}),\ \forall c \in \mathcal{V},\ \forall c' \in d(c),\ t = 1, \cdots, T \tag{2.1h}$$

$$n_{ct+1} = n_{ct} + \sum_{c' \in p(c)} y_{c't} - y_{ct},\ \forall c \in \mathcal{C}/\mathcal{O}/\mathcal{I},\ t = 1, \cdots, T \tag{2.1i}$$

$$n_{ct+1} = n_{ct} + D_{ct} - y_{ct},\ \forall c \in \mathcal{O},\ t = 1, \cdots, T \tag{2.1j}$$

$$n_{ct+1} = n_{ct} + \sum_{c' \in p(c)} \beta_{c'c} y_{c't} - y_{ct},\ \forall c \in \mathcal{I},\ t = 1, \cdots, T \tag{2.1k}$$

$$n_{c1} = n_c^{\text{init}},\ \forall c \in \mathcal{C} \tag{2.1l}$$

$$y_{ct} \geq 0,\ n_{ct} \geq 0,\ \forall c \in \mathcal{C},\ t = 1, \cdots, T \tag{2.1m}$$

$$-U \cdot z_{1ijj't} \leq b_{ijj'} - t \leq U(1 - z_{1ijj't}) - \epsilon,\ \forall i \in \mathcal{R},\ \forall j \in \mathcal{F},\ j' = 1, \cdots, N_{\text{cy}},\ t = 1, \cdots, T \tag{2.1n}$$

$$-U \cdot z_{2ijj't} + \epsilon \leq t - e_{ijj'} \leq U(1 - z_{2ijj't}),\ \forall i \in \mathcal{R},\ \forall j \in \mathcal{F},\ j' = 1, \cdots, N_{\text{cy}},\ t = 1, \cdots, T \tag{2.1o}$$

$$\sum_{j \in \mathcal{F}} (z_{1ijj't} + z_{2ijj't}) \leq |\mathcal{F}| + 1,\ \forall i \in \mathcal{R},\ j' = 1, \cdots, N_{\text{cy}},\ t = 1, \cdots, T \tag{2.1p}$$

$$o_i \leq l_i,\ \forall i \in \mathcal{R} \tag{2.1q}$$

$$b_{i1j'} = l_i \cdot (j' - 1) - o_i,\ \forall i \in \mathcal{R},\ j' = 1, \cdots, N_{\text{cy}} \tag{2.1r}$$

$$e_{ijj'} = b_{ijj'} + g_{ij},\ \forall i \in \mathcal{R},\ \forall j \in \mathcal{F},\ j' = 1, \cdots, N_{\text{cy}} \tag{2.1s}$$

$$b_{ijj'} = e_{ij-1j'},\ \forall i \in \mathcal{R},\ \forall j \in \mathcal{F}/\{1\},\ j' = 1, \cdots, N_{\text{cy}} \tag{2.1t}$$

$$l_i = \sum_{j \in \mathcal{F}} g_{ij},\ \forall i \in \mathcal{R} \tag{2.1u}$$

$$G_{\min} \leq g_{ij} \leq G_{\max},\ \forall i \in \mathcal{R},\ \forall j \in \mathcal{F} \tag{2.1v}$$

$$z_{1ijj't},\ z_{2ijj't} \in \{0, 1\},\ \forall i \in \mathcal{R},\ \forall j \in \mathcal{F},\ j' = 1, \cdots, N_{\text{cy}},\ t = 1, \cdots, T \tag{2.1w}$$

The first term of the objective function (2.1a) is to maximize the summation of the throughput of the network over all time steps. The second term of (2.1a), defined as the CTM objective term, is to force y_{ct} for each cell $c \in \mathcal{C}$ at each time step $t \in \{1, \cdots, T\}$ to obtain the minimum of the right-hand sides of constraints (2.1b)–(2.1h), so as to ensure all vehicles to travel forward as much as possible, where y_{ct} is weighted by the remaining $T - t$ time steps. The parameter α is set to balance between these two objectives, where a larger α can eliminate the vehicle-holding problem to a certain extent.

Constraints (2.1b)–(2.1m) establish the fundamental relationships in CTM. Specifically, constraints (2.1b) indicate that the number of vehicles leaving a cell c is limited by the number of vehicles inside cell c. Constraints (2.1c)–(2.1d) indicate that the number of vehicles leaving a cell c is limited by the flow capacity of cell c. Notice that for an intersection cell, its capacity is determined by the related traffic signal, which means that if the related traffic signal is red, the capacity should be zero. Constraints (2.1e)–(2.1f) indicate that the number of vehicles leaving a cell c is also limited by the flow capacity of its processing cell c'. Notice that a diverging cell has more than one processing cell and the number of vehicles entering each processing cell is estimated by the turning ratio. Constraints (2.1g)–(2.1h) indicate that the number of vehicles leaving a cell c should be limited by the number of vehicles that can enter its processing cells $d(c)$. Constraints (2.1i)–(2.1k) are flow conservation equations for cells, implying that the difference between the number of vehicles in a cell c between two consecutive time steps t and $t + 1$ equals to the number of vehicles coming from preceding cells minus the number of vehicles leaving cell c during time $[t, t + 1)$. Notice that the number of vehicles entering each origin cell is the traffic demand. Constraints (2.1l) are the initial condition of the number of vehicles inside

each cell. Constraints (2.1m) indicate that the number of vehicles leaving and inside each cell should be non-negative.

Constraints (2.1n)–(2.1w) are related to signal timing decisions. Constraints (2.1n)–(2.1o) describe the relationship between each time step t, the start time b and the end time e of the time interval when the traffic signal of each phase is green. For each intersection $i \in \mathcal{R}$, each phase $j \in \mathcal{F}$ and each cycle $j' \in \{1, \cdots, N_{\text{cy}}\}$ at each time step t, $z_{1ijj't} = 1$ if $t \geq b_{ijj'}$ and $z_{1jj't} = 0$ otherwise. Similarly, $z_{2ijj't} = 1$ if $t \leq e_{ijj't}$ and $z_{2ijj't} = 0$ otherwise. Constraints (2.1p) indicate that each time step, there is only one phase with green light. Constraints (2.1q) indicate that the offset should be less than the cycle length. Constraints (2.1r)–(2.1t) detail the steps for computing the start and the end of the time interval at each cycle when the traffic signal of each phase is green based on the cycle length and the length of the time interval. Constraints (2.1u) indicate that the sum of the green time over all phases should be equal to the cycle length. Constraints (2.1v) bound the green time from below and above. Constraints (2.1w) require z_1 and z_2 being binary variables.

2.3.3 Stochastic Optimization Model

In a real traffic network, due to unpredictable driving behavior and lack of detectors, the demand D and turning ratios β are often random. They also vary significantly even for the same time period during weekdays versus weekends. Considering the uncertainties of parameters D and β, we extend the deterministic optimization model (2.1) to a two-stage stochastic optimization model described as follows. We use parameter ξ to denote the overall vector of uncertain parameters (i.e., $\xi = (D, \beta)$) and let P be its probability distribution. In this paper, we consider discrete distribution P and a finite set of realizations $\Xi = \{\xi^1, \ldots, \xi^K\}$ of the random vector ξ such that each scenario ξ^k has a probability p^k and $\sum_{k=1}^{K} p^k = 1$. (Note that we assume a joint distribution of parameters D and β without loss of generality, while the two can be independently distributed but we can use the same sampling-based formulation.) The resulting problem with the constructed scenarios is called the Sample Average Approximation (SAA) problem (Shapiro and Homem-de Mello, 2000). In the first stage, we optimize the traffic signal related decisions, such as the cycle length l, offset o, start and end of green time intervals b, e and binary variables z_1, z_2. For each realized sample of the uncertain parameters ξ^k, we can build a linear program to determine the number of vehicles leaving and inside cells y, n, and denote their values as D^k, β^k, y^k, n^k correspondingly. The overall stochastic MIP is presented as follows.

$$\min \quad -\sum_{k=1}^{K} p^k \Big(\sum_{c\in\mathcal{D}}\sum_{t=1}^{T} n_{ct}^k + \alpha \sum_{c\in\mathcal{C}}\sum_{t=1}^{T}(T-t)y_{ct}^k\Big) \tag{2.2a}$$

s.t. Constraints (2.1n)–(2.1w)

$$y_{ct}^k \leq n_{ct}^k,\ \forall c \in \mathcal{C},\ t = 1, \cdots, T,\ k = 1, \cdots, K \tag{2.2b}$$

$$y_{ct}^k \leq Q_{ct},\ \forall c \in \mathcal{E} \cup \mathcal{O} \cup \mathcal{M} \cup \mathcal{V},\ t = 1, \cdots, T,\ k = 1, \cdots, K \tag{2.2c}$$

$$y_{ct}^k \leq \sum_{j'=1}^{N_{\text{cy}}} (z_{1ijj't} + z_{2ijj't} - 1)Q_{ct},\ \forall c \in \mathcal{I}_{ij},\ \forall i \in \mathcal{R},\ \forall j \in \mathcal{F},\ t = 1, \cdots, T,\ k = 1, \cdots, K \tag{2.2d}$$

$$y_{ct}^k \leq Q_{c't},\ \forall c \in \mathcal{C}/\mathcal{V},\ \forall c' \in d(c),\ t = 1, \cdots, T,\ k = 1, \cdots, K \tag{2.2e}$$

$$\beta_{cc'}^k y_{ct}^k \leq Q_{c't},\ \forall c \in \mathcal{V},\ \forall i \in \mathcal{R},\ \forall j \in \mathcal{F},\ \forall c' \in d(c) k I_{ij},\ t = 1, \cdots, T,\ k = 1, \cdots, K \tag{2.2f}$$

$$y_{ct}^k \leq W_{c't}(N_{c't} - n_{c't}^k),\ \forall c \in \mathcal{C}/\mathcal{V},\ \forall c' \in d(c),\ t = 1, \cdots, T,\ k = 1, \cdots, K \tag{2.2g}$$

$$\beta_{cc'} y_{ct}^k \leq W_{c't}(N_{c't} - n_{c't}^k),\ \forall c \in \mathcal{V},\ \forall c' \in d(c),\ t = 1, \cdots, T,\ k = 1, \cdots, K \tag{2.2h}$$

$$n_{ct+1}^k = n_{ct}^k + \sum_{c'\in p(c)} y_{c't}^k - y_{ct}^k,\ \forall c \in \mathcal{C}/\mathcal{O}/\mathcal{I},\ t = 1, \cdots, T,\ k = 1, \cdots, K \tag{2.2i}$$

$$n_{ct+1}^k = n_{ct}^k + D_{ct}^k - y_{ct}^k,\ \forall c \in \mathcal{O},\ t = 1, \cdots, T,\ k = 1, \cdots, K \tag{2.2j}$$

$$n_{ct+1}^k = n_{ct}^k + \sum_{c'\in p(c)} \beta_{c'c}^k y_{c't}^k - y_{ct}^k,\ \forall c \in \mathcal{I},\ t = 1, \cdots, T,\ k = 1, \cdots, K \tag{2.2k}$$

$$n^k_{c1} = n^{\text{init}}_c,\ \forall c \in \mathcal{C},\ k = 1, \cdots, K \tag{2.2l}$$

$$y^k_{ct} \geq 0,\ n^k_{ct} \geq 0,\ \forall c \in \mathcal{C},\ t = 1, \cdots, T,\ k = 1, \cdots, K \tag{2.2m}$$

The objective function is the expected second-stage cost. Constraints (2.1n)–(2.1w) do not involve uncertain parameters and only involve first-stage planning decision variables. Constraints (2.2b)–(2.2m) are the copies of constraints (2.1b)–(2.1m) corresponding to each scenario ξ^k for $k = 1, \cdots, K$, and involve both first-stage planning variables and second-stage recourse variables.

2.4 Decomposition Algorithms

Both the deterministic and stochastic optimization models developed in Section 2.3.3 are large-scale MIPs with complex constraints, and are difficult to solve directly using off-the-shelf solvers as we will demonstrate later in numerical studies. The scalability issue mainly comes from the number of intersections and time steps. In this section, we propose efficient algorithms for solving the stochastic program (2.2) using Benders decomposition algorithm combined with spatial and temporal decomposition. We describe the Benders decomposition algorithm in Section 2.4.1, propose a spatially decentralized Benders algorithm in Section 2.4.2, and an ADMM-based spatially decentralized Benders algorithm in Section 2.4.3. Furthermore, we apply temporal decomposition to these algorithm variants in Section 2.4.4 to accelerate their computation.

2.4.1 Benders Decomposition Algorithm

We first create new variables $\theta^k,\ k = 1, \cdots, K$ and define a relaxed master problem as the first-stage problem as follows.

$$(\textbf{RMP})\ \min \quad \sum_{k=1}^{K} p^k \theta^k \tag{2.3a}$$

$$\text{s.t.} \quad \text{Constraints (2.1n)–(2.1w)}$$

$$(z_1, z_2, \theta) \in \Sigma(z_1, z_2, \theta) \tag{2.3b}$$

where $\Sigma(z_1, z_2, \theta)$ is the set of Benders cuts as linear functions of z_1, z_2 generated up to the current iteration. Given first-stage integer solutions $\hat{z}_1, \hat{z}_2$, for $k = 1, \cdots, K$, the second-stage problem for a scenario ξ^k is defined as

$$\min \quad -\sum_{c \in \mathcal{D}} \sum_{t=1}^{T} n^k_{ct} - \alpha \sum_{c \in \mathcal{C}} \sum_{t=1}^{T} (T - t) y^k_{ct} \tag{2.4a}$$

$$\text{s.t.} \quad \text{Constraints (2.2b)–(2.2m) for the scenario } \xi^k$$

$$z_1 = \hat{z}_1,\ z_2 = \hat{z}_2 \tag{2.4b}$$

We first show that the second-stage problem (2.4) is always feasible in the following theorem.

Theorem 1. *Given non-negative parameters Q, N, D, n^{init}, W and positive parameter ϵ, the feasibility set composed by constraints* (2.2b)–(2.2m) *is always non-empty given any first-stage solutions $\hat{z}_1, \hat{z}_2$.*

Proof. We firstly prove that $\hat{z}_{1ijj't} + \hat{z}_{2ijj't} \geq 1$ for each $i \in \mathcal{R}$, $j \in \mathcal{F}$, $j' = 1, \cdots, N_{\text{cy}}$ and $t = 1, \cdots, T$. Assume that there exists $\hat{z}_{1ijj't} = \hat{z}_{2ijj't} = 0$, then from constraints (2.1n)–(2.1o), we have $t \leq b_{ijj'}$ and $t \geq e_{ijj'} + \epsilon$, which contradicts $b_{ijj'} \leq e_{ijj'}$ derived by constraints (2.1q)–(2.1v). Therefore, $y^k = n^k = \mathbf{0}$ is always a feasible solution. □

From Theorem 1, we only need to add optimality cuts to the set Σ, which we will construct in the next theorem.

Theorem 2. *Associating dual variables $\rho^k_{ct},\ c \in \mathcal{C},\ t = 1, \cdots, T,\ k = 1, \cdots, K$ to constraints* (2.2b), *$\sigma^k_{ct},\ c \in \mathcal{C},\ t = 1, \cdots, T,\ k = 1, \cdots, K$ to constraints* (2.2c)–(2.2d), *$\pi^k_{cc't},\ c \in \mathcal{C},\ c' \in d(c),\ t = 1, \cdots, T,\ k =$*

$1, \cdots, K$ *to constraints* (2.2e)–(2.2f), $\gamma^k_{cc't}$, $c \in \mathcal{C}$, $c' \in d(c)$, $t = 1, \cdots, T$, $k = 1, \cdots, K$ *to constraints* (2.2g)–(2.2h), δ^k_{ct}, $c \in \mathcal{C}$, $t = 1, \cdots, T$, $k = 1, \cdots, K$ *to constraints* (2.2i)–(2.2k), *and* τ^k_c, $c \in \mathcal{C}$, $k = 1, \cdots, K$ *to constraints* (2.2l), *for* $k = 1, \cdots, K$, *the optimality cut for a scenario* ξ^k *is given by*

$$\begin{aligned}\theta^k \geq & \sum_{c\in\mathcal{C}/\mathcal{I}}\sum_{t=1}^{T} Q_{ct}\sigma^k_{ct} + \sum_{i\in\mathcal{R}}\sum_{j\in\mathcal{F}}\sum_{c\in\mathcal{I}_{ij}}\sum_{j'=1}^{N_{cy}}\sum_{t=1}^{T}(z_{1ijj't} + z_{2ijj't} - 1)Q_{ct}\sigma^k_{ct} \\ & + \sum_{c\in\mathcal{C}}\sum_{c'\in d(c)}\sum_{t=1}^{T} Q_{c't}\pi^k_{cc't} + \sum_{c\in\mathcal{C}}\sum_{c'\in d(c)}\sum_{t=1}^{T} W_{c't}N_{c't}\gamma^k_{cc't} \\ & + \sum_{c\in\mathcal{O}}\sum_{t=1}^{T} D^k_{ct}\delta^k_{ct} + \sum_{c\in\mathcal{C}} n^{init}_c\tau^k_c \end{aligned} \tag{2.5}$$

Proof. The proof follows directly the strong duality of the linear programming models with fixed binary-valued inputs $\hat{z}_1$ and $\hat{z}_2$. □

For every iteration, we solve (**RMP**) with current Benders' cuts and obtain feasible solutions $\hat{z}_1, \hat{z}_2$. The optimal value of (**RMP**) provides a lower bound of the original stochastic program (2.2). Then we compute optimal dual solutions to the second-stage problems (2.4) for all scenarios, and the expectation of their optimal values provides an upper bound of the original stochastic program (2.2). When the gap between the upper bound and the lower bound is closed, the algorithm can be terminated.

2.4.2 Spatially Decentralized Benders Algorithm

Next, we consider reformulating the relaxed master problem (2.3) in a distributed way by spatial decomposition. We partition the network into N_I areas where each area contains only one intersection. For each area with an intersection i for $i \in \mathcal{R}$, let $\mathcal{C}_i, \mathcal{E}_i, \mathcal{O}_i, \mathcal{D}_i, \mathcal{M}_i, \mathcal{V}_i$ be the corresponding sets of all cells, ordinary cells, origin cells, destination cells, merge cells, and diverge cells, respectively. Our goal is to solve the relaxed master problem for each area separately which only contains an intersection i where $i \in \mathcal{R}$.

Observation 1. *By defining variables* θ^k_i *for each scenario* ξ^k, $k = 1, \cdots, K$ *and each intersection* $i \in \mathcal{R}$, *the initial form of the first-stage problem* (***RMP***) *can be reformulated as* N_I *subproblems corresponding to each intersection* $i \in \mathcal{R}$ *where each subproblem admits the following formulation.*

$$(\boldsymbol{RMP}_i) \quad \min \quad \sum_{k=1}^{K} p^k\theta^k_i \tag{2.6a}$$

$$\text{s.t.} \quad \textit{Constraints (2.1n)–(2.1w) corresponding to the intersection } i$$

$$(z_{1i}, z_{2i}, \theta_i) \in \Sigma_i(z_{1i}, z_{2i}, \theta_i) \tag{2.6b}$$

where $\Sigma_i(z_{1i}, z_{2i}, \theta_i)$ *is the set of cuts as linear functions of* z_{1i}, z_{2i} *generated up to the current iteration.*

This observation comes directly from the observation that signal constraints (2.1n)–(2.1w) can be written separately for each intersection.

Observation 2. *For* $k = 1, \cdots, K$, *the objective function of the dual problem* ($\boldsymbol{SP}^k$) *of the second-stage problem can be written as the summation of functions corresponding to each intersection* $i \in \mathcal{R}$.

To validate Observation 2, for each $k = 1, \cdots, K$ and intersection $i \in \mathcal{R}$, define function

$$\begin{aligned} F^k_{Di}(z_{1i}, z_{2i}, \rho^k, \sigma^k, \pi^k, \gamma^k, \delta^k, \tau^k) = & \sum_{c\in\mathcal{C}_i/\mathcal{I}_i}\sum_{t=1}^{T} Q_{ct}\sigma^k_{ct} + \sum_{j\in\mathcal{F}}\sum_{c\in\mathcal{I}_{ij}}\sum_{j'=1}^{N_{cy}}\sum_{t=1}^{T}(z_{1ijj't} + z_{2ijj't} - 1)Q_{ct}\sigma^k_{ct} \\ & + \sum_{c\in\mathcal{C}_i}\sum_{c'\in d(c)}\sum_{t=1}^{T} Q_{c't}\pi^k_{cc't} + \sum_{c\in\mathcal{C}_i}\sum_{c'\in d(c)}\sum_{t=1}^{T} W_{c't}N_{c't}\gamma^k_{cc't} \end{aligned}$$

$$+ \sum_{c\in\mathcal{O}_i}\sum_{t=1}^{T} D_{ct}^k \delta_{ct}^k + \sum_{c\in\mathcal{C}_i} n_c^{\text{init}} \tau_c^k. \tag{2.7}$$

Then it can be easily seen that

$$F_D^k(z_1, z_2, \rho^k, \sigma^k, \pi^k, \gamma^k, \delta^k, \tau^k) = \sum_{i\in\mathcal{R}} F_{Di}^k(z_{1i}, z_{2i}, \rho^k, \sigma^k, \pi^k, \gamma^k, \delta^k, \tau^k). \tag{2.8}$$

For each iteration, we still solve dual problems $(\mathbf{SP}^k)$ for $k = 1, \cdots, K$, but add cuts to the set Σ_i for each subproblem $(\mathbf{RMP}_i)$, $\forall i \in \mathcal{R}$ separately. Based on Observation 2, we derive an optimality cut as $\theta_i^k \geq F_{Di}^k(z_{1i}, z_{2i}, \hat{\rho}^k, \hat{\sigma}^k, \hat{\pi}^k, \hat{\gamma}^k, \hat{\delta}^k, \hat{\tau}^k)$ for the set Σ_i. The procedure of the spatially decentralized algorithm is described as follows in Algorithm 1.

Algorithm 1: Spatially Decentralized Benders Algorithm (Benders-D).

for $i \in \mathcal{R}$ **do**
  Initialize $(\mathbf{RMP}_i)$ with $\Sigma_i = \emptyset$ and $\theta_i = 0$.
while *the termination criteria is not satisfied* **do**
  for $i \in \mathcal{R}$ **do**
    Solve $(\mathbf{RMP}_i)$ to obtain optimal solution $(\hat{z}_{1i}, \hat{z}_{2i}, \hat{\theta}_i)$.
  for $k = 1, \cdots, K$ **do**
    Solve $(\mathbf{SP}^k)$ to obtain optimal solution $(\hat{\rho}^k, \hat{\sigma}^k, \hat{\pi}^k, \hat{\gamma}^k, \hat{\delta}^k, \hat{\tau}^k)$. **for** $i \in \mathcal{R}$ **do**
      if $\hat{\theta}_i^k < F_{Di}^k(\hat{z}_{1i}, \hat{z}_{2i}, \hat{\rho}^k, \hat{\sigma}^k, \hat{\pi}^k, \hat{\gamma}^k, \hat{\delta}^k, \hat{\tau}^k)$ **then**
        Add an optimality cut $\theta_i^k \geq F_{Di}^k(z_{1i}, z_{2i}, \hat{\rho}^k, \hat{\sigma}^k, \hat{\pi}^k, \hat{\gamma}^k, \hat{\delta}^k, \hat{\tau}^k)$ to the set Σ_i.
Return the objective value as $\sum_{k=1}^K p^k \sum_{i\in\mathcal{R}} \hat{\theta}_i^k$ and the solutions of $(\mathbf{RMP}_i)$, $i \in \mathcal{R}$.

Theorem 3. *Let ALG_B be the objective value obtained from Benders decomposition algorithm, and ALG_{DB} be the objective value obtained from the spatially decentralized Benders algorithm (Algorithm 1), then we have $ALG_B \leq ALG_{DB}$.*

Proof. The Benders cut (2.5) is equivalent to

$$\sum_{i\in\mathcal{R}} \theta_i^k \geq \sum_{i\in\mathcal{R}} F_{Di}^k(z_{1i}, z_{2i}, \rho^k, \sigma^k, \pi^k, \gamma^k, \delta^k, \tau^k),\ k = 1, \cdots, K. \tag{2.9}$$

Thus, from the construction of Σ and Σ_i, $i = 1, \cdots, N_I$, we know that $\prod_{i\in\mathcal{R}} \Sigma_i \subseteq \Sigma$. Recall that ALG_B is the optimal value of the $(\mathbf{RMP})$ and ALG_DB is the summation of the optimal values of all the $(\mathbf{RMP}_i)$. It follows that $\text{ALG}_\text{B} \leq \text{ALG}_\text{DB}$. □

2.4.3 ADMM-based Spatially Decentralized Benders Algorithm

The CPU time of solving the second-stage problems in the stochastic program increases significantly as network sizes increase. We further propose a distributed algorithm that solves first-stage problems as well as second-stage problems separately for each intersection based on the partition of the network in Section 2.4.2. We firstly propose a distributed formulation of second-stage problems in Section 2.4.3. Then we solve the second-stage problems for each intersection individually by applying ADMM (Boyd et al., 2011), which is described in Section 2.4.3. Furthermore, we generate optimality cuts based on the optimal value and solutions obtained by ADMM and prove that the objective value obtained by our proposed algorithm converges to the optimal value in Section 2.4.3.

Distributed Formulation

For each intersection $i \in \mathcal{R}$, we partition the set of cells $\mathcal{C}_i$ into two parts – one consists of all the cells where the constraints of these cells are related to cells in other intersections, called boundary cells and

the other consists of all the remaining cells, called internal cells (Timotheou et al., 2014). For each area containing an intersection $i \in \mathcal{R}$, let $\mathcal{B}_i^I$ be the set of input boundary cells receiving inflow traffic from a cell of a neighboring area, and $\mathcal{B}_i^O$ be the set of output boundary cells sending outflow traffic to a cell of a neighboring area. In a centralized stochastic programming model, constraints (2.2g) and (2.2i) are related to boundary cells. We rewrite these constraints separately for boundary cells and internal cells. Notice that in our grid network setting, for each boundary cell $c \in \mathcal{B}_i^O \cup \mathcal{B}_i^I$, there is only one cell receiving the inflow traffic from c and one cell sending the outflow traffic to c, denoted as d_c and p_c, indicating that $d(c) = \{d_c\}$ and $p(c) = \{p_c\}$.

For each intersection $i \in \mathcal{R}$, the constraints related to boundary cells include variables corresponding to other intersections. For each $k = 1, \cdots, K$ and $i \in \mathcal{R}$, we introduce vectors $\tilde{y}_i^k \in \mathbb{R}^{|\mathcal{B}_i^I| \times T}$ and $\tilde{n}_i^k \in \mathbb{R}^{|\mathcal{B}_i^O| \times T}$ to estimate the corresponding y^k and n^k of cells of neighboring intersections. Constraints (2.1g)–(2.1h) are rewritten as equality constraints by defining auxiliary variables $s_{ct}^k \geq 0$ for each cell $c \in \mathcal{C}$, each time step $t = 1, \cdots, T$ and each $k = 1, \cdots, K$. Let s^k be the vector form of the auxiliary variables. The second-stage problem for scenario ξ^k can be written as a distributed formulation as follows.

$$(\textbf{SP-D}^k) \ \min \quad -\sum_{c\in\mathcal{D}}\sum_{t=1}^{T} n_{ct}^k - \alpha \sum_{c\in\mathcal{C}}\sum_{t=1}^{T}(T-t)y_{ct}^k \tag{2.10a}$$

$$\text{s.t.} \quad \text{Constraints (2.2b)–(2.2f), (2.2i)–(2.2m) for the scenario } \xi^k, \ \forall c \in \mathcal{C}_i / \mathcal{B}_i^I / \mathcal{B}_i^O \tag{2.10b}$$

$$s_{ct}^k \geq 0, \ \forall c \in \mathcal{C}, \ \forall c' \in d(c), \ t = 1, \cdots, T \tag{2.10c}$$

$$y_{ct}^k + s_{ct}^k = W_{c't}(N_{c't} - n_{c't}^k), \ \forall c \in \mathcal{C}/\mathcal{V}/\mathcal{B}_i^O, \ \forall c' \in d(c), \ t = 1, \cdots, T \tag{2.10d}$$

$$\beta_{cc'} y_{ct}^k + s_{ct}^k = W_{c't}(N_{c't} - n_{c't}^k), \ \forall c \in \mathcal{V}, \ \forall c' \in d(c), \ t = 1, \cdots, T \tag{2.10e}$$

$$y_{ct}^k + s_{ct}^k = W_{d_c t}(N_{d_c t} - \tilde{n}_{ict}^k), \ \forall i \in \mathcal{R}, \ \forall c \in \mathcal{B}_i^O, \ t = 1, \cdots, T \tag{2.10f}$$

$$n_{ct+1}^k = n_{ct}^k + \tilde{y}_{ict}^k - y_{ct}^k, \ \forall i \in \mathcal{R}, \ \forall c \in \mathcal{B}_i^I, \ t = 1, \cdots, T \tag{2.10g}$$

$$\tilde{y}_{ict}^k = y_{p_c t}^k, \ \forall i \in \mathcal{R}, \ \forall c \in \mathcal{B}_i^I, \ t = 1, \cdots, T \tag{2.10h}$$

$$\tilde{n}_{ict}^k = n_{d_c t}^k, \ \forall i \in \mathcal{R}, \ \forall c \in \mathcal{B}_i^O, \ t = 1, \cdots, T \tag{2.10i}$$

Constraints (2.10b) correspond to constraints (2.2b)–(2.2f), (2.2i)–(2.2m) for the internal cells under scenario ξ^k. Constraints (2.10c) indicate that s^k should be non-negative. Constraints (2.10d)–(2.10e) refer to the equality form of constraints (2.2g)–(2.2h) under scenario ξ^k. Constraints (2.10f) and (2.10g) refer to the equality form of constraints (2.2g) and (2.2i) related to boundary cells under scenario ξ^k. Constraints (2.10h) and (2.10i) indicate that for each intersection $i \in \mathcal{R}$, $\tilde{y}_i^k$ and $\tilde{n}_i^k$ should be equal to the value of y^k and n^k of cells of neighboring intersections receiving flow from or sending flow to boundary cells.

ADMM for Second-stage Problems

In the distributed formulation, each intersection is considered as a block and the model contains several additional coupled (linear) constraints between pairs of intersection blocks. Then we can apply ADMM to solve each second-stage problem $(\textbf{SP-D}^k)$, $\forall k = 1, \cdots, K$.

Theorem 4. *For each $k = 1, \cdots, K$, the distributed second-stage problem $(\textbf{SP-D}^k)$ can be reformulated as a series of subproblems for each intersection $i \in \mathcal{R}$.*

Proof. We define $\mathbf{X}_i$ as the convex feasible region of $[y_i^k, n_i^k, s_i^k]$ defined by constraints (2.10b)–(2.10e) for each intersection $i \in \mathcal{R}$. Introducing dual variables κ_{ict}^k, $i \in \mathcal{R}$, $c \in \mathcal{B}_i^O$, $t = 1, \cdots, T$ for constraints (2.10f), dual variables λ_{ict}^k, $i \in \mathcal{R}$, $c \in \mathcal{B}_i^I$, $t = 1, \cdots, T$ for constraints (2.10g), dual variables μ_{ict}^k, $i \in \mathcal{R}$, $c \in \mathcal{B}_i^I$, $t = 1, \cdots, T$ for constraints (2.10h), and dual variables ν_{ict}^k, $i \in \mathcal{R}$, $c \in \mathcal{B}_i^O$, $t = 1, \cdots, T$ for constraints (2.10i), we use $\kappa_i^k, \lambda_i^k, \mu_i^k, \nu_i^k$ to represent vector forms of these dual variables and define

an augmented dual Lagrangian function with a feasible set $\mathbf{X}_i$ for each intersection $i \in \mathcal{R}$ as follows.

$$\begin{aligned}\mathcal{L}_i^k(y_i^k, n_i^k, s_i^k, \tilde{y}_i^k, \tilde{n}_i^k, \kappa_i^k, \lambda_i^k, \mu_i^k, \nu_i^k) = &- \sum_{c\in\mathcal{D}_i}\sum_{t=1}^{T} n_{ct}^k - \alpha \sum_{c\in\mathcal{C}_i}\sum_{t=1}^{T}(T-t)y_{ct}^k \\ &+ \sum_{c\in\mathcal{B}_i^O}\sum_{t=1}^{T}\kappa_{ict}^k(y_{ct}^k + s_{ct}^k - W_{d_ct}(N_{d_ct} - \tilde{n}_{ict}^k)) \\ &+ \sum_{c\in\mathcal{B}_i^I}\sum_{t=1}^{T}\lambda_{ict}^k(n_{ct+1}^k - n_{ct}^k - \tilde{y}_{ict}^k + y_{ct}^k) \\ &+ \sum_{c\in\mathcal{B}_i^I}\sum_{t=1}^{T}\mu_{ict}^k(\tilde{y}_{ict}^k - y_{p_ct}^k) + \sum_{c\in\mathcal{B}_i^O}\sum_{t=1}^{T}\nu_{ict}^k(\tilde{n}_{ict}^k - n_{d_ct}^k) \\ &+ \frac{L}{2}\sum_{c\in\mathcal{B}_i^O}\sum_{t=1}^{T}\|y_{ct}^k + s_{ct}^k - W_{d_ct}(N_{d_ct} - \tilde{n}_{ict}^k)\|^2 \\ &+ \frac{L}{2}\sum_{c\in\mathcal{B}_i^I}\sum_{t=1}^{T}\|n_{ct+1}^k - n_{ct}^k - \tilde{y}_{ict}^k + y_{ct}^k\|^2 \\ &+ \frac{L}{2}\sum_{c\in\mathcal{B}_i^I}\sum_{t=1}^{T}\|\tilde{y}_{ict}^k - y_{p_ct}^k\|^2 + \frac{L}{2}\sum_{c\in\mathcal{B}_i^O}\sum_{t=1}^{T}\|\tilde{n}_{ict}^k - n_{d_ct}^k\|^2, \end{aligned} \tag{2.11}$$

where L is the Lagrangian penalty parameter. The problem (**SP-D**^{k}) can be rewritten as the minimization problem of the augmented Lagrangian function with a feasible set composed by constraints (2.10b)–(2.10g). Following the definitions of dual variables, the augmented Lagrangian function is the summation of functions $\mathcal{L}_i^k$ for all intersections $i \in \mathcal{R}$, i.e.,

$$\mathcal{L}^k(y^k, n^k, s^k, \tilde{y}^k, \tilde{n}^k, \kappa^k, \lambda^k, \mu^k, \nu^k) = \sum_{i\in\mathcal{R}} \mathcal{L}_i^k(y_i^k, n_i^k, s_i^k, \tilde{y}_i^k, \tilde{n}_i^k, \kappa_i^k, \lambda_i^k, \mu_i^k, \nu_i^k) \tag{2.12}$$

The minimization problem of $\mathcal{L}^k$ is equivalent to a series of problems that minimize $\mathcal{L}_i^k$ for each intersection $i \in \mathcal{R}$. □

For each intersection $i \in \mathcal{R}$, ADMM consists of three main steps: (i) updating variables y_i^k, n_i^k, s_i^k, (ii) updating variables $\tilde{y}_i^k, \tilde{n}_i^k$, and (iii) updating dual variables $\kappa_i^k, \lambda_i^k, \mu_i^k, \nu_i^k$. Specifically, in iteration $l+1$, we update variables based on current values $y^{kl}, n^{kl}, s^{kl}, \tilde{y}^{kl}, \tilde{n}^{kl}, \kappa^{kl}, \lambda^{kl}, \mu^{kl}, \nu^{kl}$ as follows.

$$[y_i^{kl+1}, n_i^{kl+1}, s_i^{kl+1}] = \underset{[y_i^k, n_i^k, s_i^k]\in\mathbf{X}_i}{\arg\min}\ \mathcal{L}_i^k(y_i^k, n_i^k, s_i^k, \tilde{y}^{kl}, \tilde{n}^{kl}, \kappa_i^{kl}, \lambda_i^{kl}\mu_i^{kl}, \nu_i^{kl}),\ \forall i \in \mathcal{R} \tag{2.13a}$$

$$[\tilde{y}_i^{kl+1}, \tilde{n}_i^{kl+1}] = \arg\min \mathcal{L}_i^k(y_i^{kl+1}, n_i^{kl+1}, s_i^{kl+1}, \tilde{y}_i^k, \tilde{n}_i^k, \kappa_i^{kl}, \lambda_i^{kl}, \mu_i^{kl}, \nu_i^{kl}),\ \forall i \in \mathcal{R} \tag{2.13b}$$

$$\kappa_{ict}^{kl+1} = \kappa_{ict}^{kl} + L(y_{ct}^{kl+1} + s_{ct}^{kl+1} - W_{d_ct}(N_{d_ct} - \tilde{n}_{ict}^{kl+1})),\ \forall i \in \mathcal{R},\ \forall c \in \mathcal{B}_i^O,\ t = 1, \cdots, T \tag{2.13c}$$

$$\lambda_{ict}^{kl+1} = \lambda_{ict}^{kl} + L(n_{ct+1}^{kl+1} - n_{ct}^{kl+1} - \tilde{y}_{ict}^{kl+1} + y_{ct}^{kl+1}),\ \forall i \in \mathcal{R},\ \forall c \in \mathcal{B}_i^I,\ t = 1, \cdots, T \tag{2.13d}$$

$$\mu_{ict}^{kl+1} = \mu_{ict}^{kl} + L(\tilde{y}_{ct}^{kl+1} - y_{p_ct}^{kl+1}),\ \forall i \in \mathcal{R},\ \forall c \in \mathcal{B}_i^I,\ t = 1, \cdots, T \tag{2.13e}$$

$$\nu_{ict}^{kl+1} = \nu_{ict}^{kl} + L(\tilde{n}_{ct}^{kl+1} - n_{d_ct}^{kl+1}),\ \forall i \in \mathcal{R},\ \forall c \in \mathcal{B}_i^O,\ t = 1, \cdots, T \tag{2.13f}$$

Since the procedure of updating variables is separable for each intersection, the computation can be conducted in parallel for different intersections at each iteration, which can speed up the computation drastically. In practice, each subproblem can be solved by local computers installed at each intersection and then communicate with each other to update the duals, to fully utilize connected transportation technologies.

ADMM-based Spatially Decentralized Benders Algorithm

We use the same formulation of $(\mathbf{RMP}_i)$ in Section 2.4.2 as first-stage problems for all the intersections $i \in \mathcal{R}$. For second-stage problems, we solve them by ADMM and obtain the optimal solutions $\hat{n}^k, \hat{y}^k$ and the optimal value $\mathcal{L}_i^{k^*}(z_{1i}, z_{2i})$ for each intersection $i \in \mathcal{R}$ and each $k = 1, \cdots, K$.

Theorem 5. *For each intersection $i \in \mathcal{R}$, Given first-stage optimal solutions $\hat{z}_{1i}$, $\hat{z}_{2i}$, the optimal value of the second-stage problem corresponding to a scenario ξ^k, $k = 1, \cdots, K$ is linear in $\hat{z}_{1i}$, $\hat{z}_{2i}$.*

Proof. For each $k = 1, \cdots, K$ and $i \in \mathcal{R}$, let $\hat{\tilde{y}}_i^k, \hat{\tilde{n}}_i^k, \hat{\kappa}_i^k, \hat{\lambda}_i^k, \hat{\mu}_i^k, \hat{\nu}_i^k$ be the optimal solutions of variables $\tilde{y}_i^k, \tilde{n}_i^k, \kappa_i^k, \lambda_i^k, \mu_i^k, \nu_i^k$. We consider the minimization problem of $\mathcal{L}_i^k(y_i^k, n_i^k, s_i^k, \hat{\tilde{y}}_i^k, \hat{\tilde{n}}_i^k, \hat{\kappa}_i^k, \hat{\lambda}_i^k, \hat{\mu}_i^k, \hat{\nu}_i^k)$ with constraints (2.10b)–(2.10c). The dual Lagrangian function of this minimization problem is in the form of

$$
\begin{aligned}
&\tilde{\mathcal{L}}_i^k(\hat{z}_{1i}, \hat{z}_{2i}, y_i^k, n_i^k, s_i^k, \rho^k, \sigma^k, \pi^k, \gamma^k, \delta^k, \tau^k) \\
=&\mathcal{L}_i^k(y_i^k, n_i^k, s_i^k, \hat{\tilde{y}}_i^k, \hat{\tilde{n}}_i^k, \hat{\kappa}_i^k, \hat{\lambda}_i^k, \hat{\mu}_i^k, \hat{\nu}_i^k) + \sum_{c \in \mathcal{C}_i / \mathcal{I}_i} \sum_{t=1}^{T} (Q_{ct} - y_{ct}^k)\sigma_{ct}^k \\
&+ \sum_{j \in \mathcal{F}} \sum_{c \in \mathcal{I}_{ij}} \sum_{j'=1}^{N_{cy}} \sum_{t=1}^{T} ((\hat{z}_{1ijj't} + \hat{z}_{2ijj't} - 1)Q_{ct} - y_{ct}^k)\sigma_{ct}^k + \sum_{c \in \mathcal{C}_i} \sum_{c' \in d(c)} \sum_{t=1}^{T} (Q_{c't} - y_{ct}^k)\pi_{cc't}^k \\
&+ \sum_{c \in \mathcal{C}_i / \mathcal{B}_i^O} \sum_{c' \in d(c)} \sum_{t=1}^{T} (W_{c't}N_{c't} - y_{ct}^k - s_{ct}^k - W_{c't}n_{ct}^k)\gamma_{cc't}^k \\
&+ \sum_{c \in \mathcal{O}_i} \sum_{t=1}^{T} (D_{ct}^k - y_{ct}^k + n_{ct}^k - n_{ct+1}^k)\delta_{ct}^k + \sum_{c \in \mathcal{C}_i} (n_c^{\text{init}} - n_{c1}^k)\tau_c^k
\end{aligned} \tag{2.14}
$$

The only term in $\tilde{\mathcal{L}}_i^k$ corresponding to $\hat{z}_{1i}, \hat{z}_{2i}$ is a linear term that is not related to y_i^k, n_i^k, s_i^k. Hence, the optimal value of the problem $\max_{\rho^k \geq 0, \sigma^k \geq 0, \pi^k \geq 0, \gamma^k, \delta^k, \tau^k} \min_{y_i^k, n_i^k, s_i^k} \tilde{\mathcal{L}}_i^k$ only includes the linear term corresponding to $\hat{z}_{1i}, \hat{z}_{2i}$. Since the minimization problem of $\mathcal{L}_i^k$ is a semi-definite quadratic program, strong duality holds. From the convergence property of ADMM, we have,

$$
\mathcal{L}_i^{k^*}(\hat{z}_{1i}, \hat{z}_{2i}) = \max_{\rho^k \geq 0, \sigma^k \geq 0, \pi^k \geq 0, \gamma^k, \delta^k, \tau^k} \min_{y_i^k, n_i^k, s_i^k} \tilde{\mathcal{L}}_i^k(\hat{z}_{1i}, \hat{z}_{2i}, [y_i^k, n_i^k, s_i^k] \in \mathbf{X}_i, \rho^k, \sigma^k, \pi^k, \gamma^k, \delta^k, \tau^k) \tag{2.15}
$$

Therefore, the optimal value of the second-stage problem $\mathcal{L}_i^{k^*}$ is linear in $\hat{z}_{1i}, \hat{z}_{2i}$. □

Proposition 1. *The feasibility set of $\rho^k, \sigma^k, \pi^k, \gamma^k, \delta^k, \tau^k$ generated by the program $\min_{y_i^k, n_i^k, s_i^k} \tilde{\mathcal{L}}_i^k$ does not depend on first-stage decision variables.*

Proof. The proof follows directly from the form of $\tilde{\mathcal{L}}_i^k$ in the proof of Theorem 5. □

For each $k = 1, \cdots, K$ and $i \in \mathcal{R}$, given the optimal value $\mathcal{L}_i^{k^*}$ and dual optimal solutions σ_{ct}^k, $c \in \mathcal{I}_{ij}$, $j \in \mathcal{F}$, $t = 1, \cdots, T$ corresponding to constraints (2.1d) for scenario ξ^k and intersection i, we can generate an optimality cut as:

$$
\begin{aligned}
\theta_i^k \geq \mathcal{L}_i^{k*} &- \sum_{j \in \mathcal{F}} \sum_{c \in \mathcal{I}_{ij}} \sum_{j'=1}^{N_{cy}} \sum_{t=1}^{T} (\hat{z}_{1ijj't} + \hat{z}_{2ijj't} - 1)Q_{ct}\hat{\sigma}_{ct}^k \\
&+ \sum_{j \in \mathcal{F}} \sum_{c \in \mathcal{I}_{ij}} \sum_{j'=1}^{N_{cy}} \sum_{t=1}^{T} (z_{1ijj't} + z_{2ijj't} - 1)Q_{ct}\hat{\sigma}_{ct}^k
\end{aligned} \tag{2.16}
$$

For simplicity, we rewrite the right-hand side of (2.16) as $\tilde{F}_{D_i}^k(z_{1i}, z_{2i}, \hat{z}_{1i}, \hat{z}_{2i}, \mathcal{L}_i^{k*}, \hat{\sigma})$. The procedure of ADMM-based spatially decentralized Benders algorithm is described in Algorithm 2.

Theorem 6. *The objective value obtained by the ADMM-based spatially decentralized Benders algorithm (i.e., Algorithm 2) equals to the optimal objective value of the stochastic programming model* (2.2).

Algorithm 2: ADMM-based Spatially Decentralized Benders Algorithm (Benders-ADMM).

for $i \in \mathcal{R}$ **do**
  Initialize (**RMP**$_i$) with $\Sigma_i = \emptyset$ and $\theta_i = 0$.
while *the termination criteria is not satisfied* **do**
  for $i \in \mathcal{R}$ **do**
    Solve (**RMP**$_i$) to obtain optimal solution $(\hat{z}_{1i}, \hat{z}_{2i}, \hat{\theta}_i)$.
  for $k = 1, \cdots, K$ **do**
    Solve (**SP-D**^{k}) by ADMM in parallel according to Section 2.4.3 and obtain the optimal value $\mathcal{L}_i^{k*}$ and optimal dual variables $\hat{\sigma}$. **for** $i \in \mathcal{R}$ **do**
      if $\hat{\theta}_i^k < \mathcal{L}_i^{k*}$ **then**
        Add optimality cut $\theta_i^k \geq \tilde{F}_{D_i}^k(z_{1i}, z_{2i}, \hat{z}_{1i}, \hat{z}_{2i}, \mathcal{L}_i^{k*}, \hat{\sigma})$ to Σ_i.
Return the objective value as $\sum_{k=1}^K p^k \sum_{i\in\mathcal{R}} \hat{\theta}_i^k$ and the solutions of (**RMP**$_i$), $i \in \mathcal{R}$.

Proof. Based on Theorem 4 and Theorem 5, the original stochastic programming model (2.2) can be reformulated as:

$$\min \quad \sum_{k=1}^{K} p^k \sum_{i\in\mathcal{R}} \theta_i^k \tag{2.17a}$$

$$\text{s.t.} \quad \text{Constraints (2.1n)–(2.1w)}$$

$$\theta_i^k \geq \max_{\rho^k\geq 0, \sigma^k \geq 0, \pi^k \geq 0, \gamma^k, \delta^k, \tau^k} \min_{y_i^k, n_i^k, s_i^k} \tilde{\mathcal{L}}_i^k(z_{1i}, z_{2i}, y_i^k, n_i^k, s_i^k, \rho^k, \sigma^k, \pi^k, \gamma^k, \delta^k, \tau^k),$$

$$\forall i \in \mathcal{R},\ k = 1, \cdots, K \tag{2.17b}$$

From Proposition 1, we know that the feasibility set composed by constraint (2.17b) is a subset of the feasibility set composed by the added optimality cuts in the ADMM-based spatially decentralized Benders algorithm. Hence the summation of the objective values of (**RMP**$_i$) for all $i \in \mathcal{R}$ provides a lower bound of the original stochastic programming model. Furthermore, the summation of $\mathcal{L}_i^{k*}$ provides an upper bound of the original stochastic programming model since it is the objective value of a feasible solution. The algorithm terminates at the optimal objective value when the gap between the lower bound and the upper bound is closed. □

Remark 1. *Spatially decentralized Benders algorithm (Algorithm 1) and ADMM-based spatially decentralized Benders algorithm (Algorithm 2) can also be employed when the partitioned areas contain multiple intersections.*

2.4.4 Temporal Decomposition

For the first-stage problems of each intersection $i \in \mathcal{R}$, the computational complexity mainly depends on the number of time steps, T.

Lemma 1. *For each $i \in \mathcal{R}$, the number of variables and constraints in (**RMP**$_i$) grows as $O(T^2)$ with the number T of time steps.*

Proof. The number of variables in (**RMP**$_i$) is $2|\mathcal{F}|N_{\text{cy}}T + 2|\mathcal{F}|N_{\text{cy}}$. As $|\mathcal{F}|$ does not depend on T and N_{cy} grows linearly with T, the number of variables grows as $O(T^2)$. Similarly, the number of constraints is $O(|\mathcal{F}|N_{\text{cy}}T)$, which also grows as $O(T^2)$. This completes the proof. □

To reduce the size of the first-stage problem, we pre-determine the cycle lengths l_i as $\hat{l}_i$ for each $i \in \mathcal{R}$ based on the traffic demand volume according to Koonce and Rodegerdts (2008). Since the traffic signal control plan is the same for each cycle, we consider (**RMP**$_i$) where the number of time steps is $\hat{l}_i$ for each intersection $i \in \mathcal{R}$. We define $t' = t \mod \hat{l}_i$ and reformulate constraints (2.2d) as:

$$y_{ct}^k \leq \sum_{j'=1}^{N_{\text{cy}}} (z_{1ijj't'} + z_{2ijj't'} - 1) Q_{ct},\ \forall c \in \mathcal{I}_{ij},\ \forall i \in \mathcal{R},\ \forall j \in \mathcal{F},\ t = 1, \cdots, T,\ k = 1, \cdots, K. \tag{2.18}$$

Lemma 2. *For each intersection $i \in \mathcal{R}$ and the corresponding first-stage problem $(\boldsymbol{RMP}_i)$, by setting $t \in \{1, \cdots, \tilde{l}_i\}$, the maximum number of cycles $N_{cy} = 2$.*

Proof. From constraints (2.1q)–(2.1u), given $l_i = \hat{l}_i$, we have

$$-\hat{l}_i \leq b_{i11} \leq 0, \ \forall i \in \mathcal{R} \tag{2.19a}$$

$$e_{i|\mathcal{F}|2} \geq \hat{l}_i, \ \forall i \in \mathcal{R} \tag{2.19b}$$

The time horizon we consider is $\{1, \cdots, \hat{l}_i\}$, which is a subset of $[b_{i11}, e_{i|\mathcal{F}|2}]$, and thus $N_{\text{cy}} = 2$. □

Theorem 7. *By pre-determining the cycle length, the problem size of $(\boldsymbol{RMP}_i)$ for each $i \in \mathcal{R}$ only grows linearly with the cycle length.*

Proof. Combining Lemma 1 and Lemma 2, it is clear that the number of variables and the number of constraints are both $O(\hat{l}_i)$. □

Because the number of variables and constraints in (**RMP**) is the summation of the number of variables and constraints in $(\mathbf{RMP}_i)$ for all $i \in \mathcal{R}$, Theorem 7 holds if we pre-determine the cycle lengths in (**RMP**).

Remark 2. *In this paper, the pre-determined cycle length for all the intersections are the same (Koonce and Rodegerdts, 2008). The cycle length l^{fix} is set as the mean value of $l_i, i \in \mathcal{R}$ computed by the following formula (Webster, 1958).*

$$l_i = \left\lceil \frac{|\mathcal{F}| * 7.5 + 5}{1 - \sum_{j=1}^{4} D_j / Q_j} \right\rceil \tag{2.20}$$

where D_j and Q_j are the traffic demand and maximum flow capacity of the direction corresponding to a phase j.

2.5 Numerical Studies

We apply algorithms proposed in Section 2.4 to solve the traffic signal control problem on instances of randomly generated grid networks and real-world traffic networks. In Section 2.5.1, we introduce the experimental design, including the warm-up initialization in Section 2.5.1, the network design of instances in Sections 2.5.1–2.5.1 and the out-of-sample evaluation in Section 2.5.1. In Section 2.5.2 and Section 2.5.3, we present the computational results to demonstrate the efficacy of our approaches.

2.5.1 Experimental Design

We firstly introduce the warm start technique for the initialization of our model. Then we introduce the network design of our instances. Lastly, we describe the evaluation metrics and computational procedures for out-of-sample tests.

Warm-up Initialization

To initialize the number of vehicles inside each cell $c \in \mathcal{C}$, we use a warm start technique according to Webster (1958). We define a fixed traffic signal time plan and compute the number of vehicles in each cell $c \in \mathcal{C}$ by solving the model with an objective function as the second term of (2.1a) and constraints (2.1b)–(2.1m). For each each phase $j \in \mathcal{F}$, the green time of a phase j is defined as

$$g_j^{\text{fix}} = \frac{D_j}{\sum_{j=1}^{4} D_j} l^{\text{fix}}, \tag{2.21}$$

where D_j is the traffic demand of the direction corresponding to a phase j and l^{fix} is the cycle length. For each cell $c \in \mathcal{C}$, we set n_c^{init} as the number of vehicles inside c at the last time step.

Random Grid Networks

We conduct numerical studies on randomly generated grid networks with the size $N_{\text{row}} \times N_{\text{col}}$ where N_{row} is the number of rows and N_{col} is the number of intersections in each row that has the same structure of intersections and road segments. We set the parameter values of the model described in Section 2.3 as follows. Let "veh" denote the number of vehicles. For each intersection cell $c \in \mathcal{I}$ and time step $t = 1, \cdots, T$, we set $Q_{ct} = 1.5$ veh and $N_{ct} = 6$ veh, meaning that at most 1.5 and 6 vehicles can flow through and reside in a cell at each time step, respectively. For the other cells $c \in \mathcal{C}/\mathcal{I}$ and time step $t = 1, \cdots, T$, $Q_{ct} = 3$ veh and $N_{ct} = 12$ veh, meaning that at most 3 and 12 vehicles can flow through and reside in a cell at each time step. The ratio of shock-wave speed over free-flow speed is the same for all the cells in $\mathcal{C}$ and all the time steps in $\{1, \cdots, T\}$, which is set as $W = 1/3$. The initialized number of vehicles n^{init} is generated by the warm-start technique described in Section 2.5.1. The minimum green time $G_{\min} = 6$ seconds and the maximum green time $G_{\max} = 75$ seconds. The whole time horizon is half an hour and has 600 time steps. We set the weight parameter $\alpha = 0.001$ in the objective function.

We generate random samples of traffic demand and turning ratios as follows. We assume uniform arrivals of vehicles during the half-an-hour time horizon, and therefore values of D_{ct} are the same for all time steps $t = 1, \cdots, T$ for any origin cell $c \in \mathcal{O}$. The traffic demand for each cell $c \in \mathcal{O}$ has a truncated Normal distribution defined on $[0, \infty)$ shown in Table 2.2 where column "SD/Mean" represents the ratio between standard deviation and the mean value. The unit of traffic demand is the number of vehicles per hour (veh/h).

Table 2.2: Distribution of randomly generated traffic demand

Distribution Instance	Mean Value (veh/h)	SD/Mean
1	200 (E-W*), 50 (S-N**)	2
2	200 (E-W), 50 (S-N)	3
3	200 (E-W), 50 (S-N)	4
4	400 (E-W), 100 (S-N)	2
5	400 (E-W), 100 (S-N)	3
6	400 (E-W), 100 (S-N)	4

*E-W: direction of east and west;
**S-N: direction of south and north;

The turning ratios of all the diverge cells $c \in \mathcal{D}$ follow truncated Normal distribution defined on $[0, 1]$ with mean values $[0.15, 0.72, 0.13]$ where the three elements represent the ratio of turning left, going straight and turning right, respectively. The ratio between the standard deviation and the mean value is set to 0.3. We test our approaches on randomly generated grid networks having sizes 4×4, 2×8, 6×6 and 10×10 and the distribution instance #4 of traffic demand given in Table 2.2. We test the other distribution instances in Table 2.2 only for the 4×4 grid network.

We generate 10 in-sample scenarios to formulate the stochastic optimization model. As a benchmark, we firstly use Gurobi to solve the stochastic programming model (2.2) directly. Then we employ Benders decomposition algorithm with and without temporal decomposition described in Section 2.4.4 to solve the model. Furthermore, based on the temporal decomposition, we apply Benders decomposition algorithm, spatially decentralized Benders algorithm, and the ADMM-based spatially decentralized Benders algorithm. For the 4×4 grid network, we build a deterministic optimization model by setting the traffic demand and turning ratios as the expectation and solve the model by applying ADMM-based spatially decentralized Benders algorithm with $K = 1$.

Real-world Traffic Networks

We also test all the proposed algorithms on a real traffic network based on the roads of Downtown in Ann Arbor (in Michigan, United States) where the network layout is shown in Figure 2.2. In the figure, the lines indicate the roads, and the bullets indicate the signalized intersections. This traffic

network contains 14 corridors, 37 signalized intersections, and 27 unsignalized intersections with road structures such as stop signs and one-way roads. All the parameters, including mean values of traffic demand and turning ratios, are computed based on collected real-world traffic data. We test an instance with high average traffic demand during peak hours where we set the SD/Mean ratio to 2 and an instance with low average traffic demand during off-peak hours where we set the SD/Mean ratio to 3. We set the ratio between standard deviation and mean of turning ratios as 0.3.

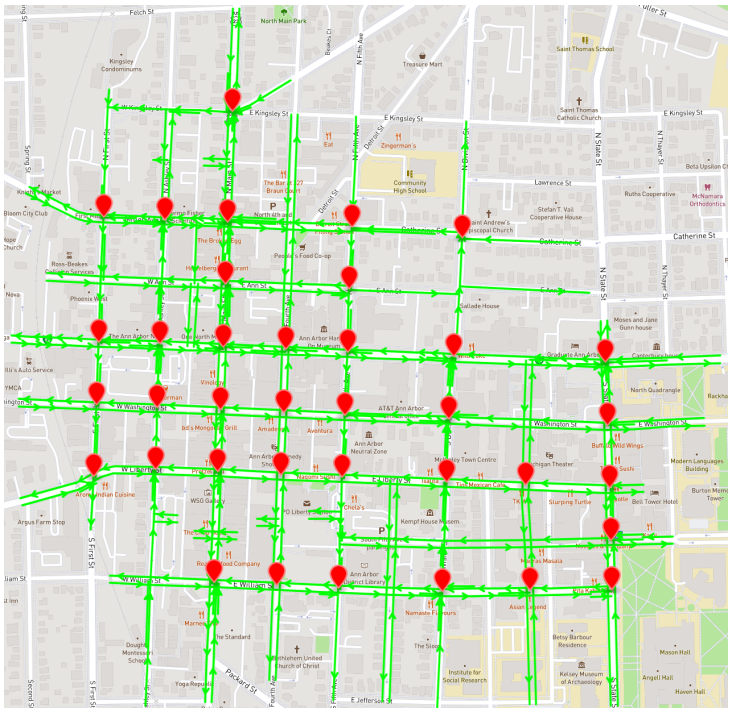


Figure 2.2: Real-world traffic network of Ann Arbor downtown with 64 intersections and 14 corridors

We generate 10 scenarios to formulate the stochastic optimization model and apply ADMM-based spatially decentralized Benders algorithm to solve the model. We also set the traffic demand and turning ratios as the mean value to build a deterministic model and apply ADMM-based spatially decentralized Benders algorithm with $K = 1$ to solve it.

Remark 3. *We set a heuristic lower bound of the green time that the green length of each phase j should be no less than 60% of g_j^{fix} and apply early termination to make the solutions more practically. Notice that our algorithm still works without these settings.*

Out-of-sample Evaluation

Metrics for Evaluation After obtaining a traffic signal control plan, we evaluate its performance by conducting CTM simulation according to Daganzo (1992). Let y_{ct}^* be the obtained solution of the number of vehicles leaving a cell $c \in \mathcal{C}$ at a time step $t = 1, \cdots, T$. We define a link as a set of cells that belong to the same road segment connecting two neighboring intersections. For each link starting from a cell c_1 and ending at a cell c_2, we define the cumulative number of vehicles at time step $t = 1, \cdots, T$ of the inflow and outflow as $CN_{\text{in}}(t)$ and $CN_{\text{out}}(t)$. The computation of these two metrics follows:

$$CN_{\text{in}}(t) = \sum_{t'=1}^{t} y_{proc(c_1)t}^* \tag{2.22a}$$

$$CN_{\text{out}}(t) = \sum_{t'=1}^{t} y_{c_2 t}^* \tag{2.22b}$$

We define the cumulative number of vehicles of outflow at time step t as $CN^*(t)$ by assuming free-flow speed, and compute the value as

$$CN^*(t) = CN_{\text{in}}(t - N_c), \tag{2.23}$$

where N_c is the number of cells contained in the link. The performance of a traffic signal control plan is evaluated for every link during the T time steps by the total travel time $\sum_{t=1}^{T}(CN_{\text{in}}(t) - CN_{\text{out}}(t))$ and the total delay $\sum_{t=1}^{T}(CN^*(t) - CN_{\text{out}}(t))$. We also compute the average travel time and the average delay of each vehicle. In addition, the total number of vehicles traveling through the traffic network during the time horizon is computed by the summation of $CN_{\text{out}}(T)$ for all the links ending at destination cells.

Out-of-sample Evaluation Procedures For both randomly generated grid networks and the real-world traffic network, we generate 5 replications of instances with the same parameter settings, each having 100 independently identically distributed scenarios with the same distribution as the one used in in-sample computation. We conduct the out-of-sample tests based on CTM simulation for different traffic signal control plans on these replications. We present the averages of the in-sample and out-of-sample objective values, the average travel delay, and the total throughput across the 500 scenarios. For the real-world traffic network, we also compare the performance with our baseline solution that sets the green time of each phase as g_j^{fix}.

All the numerical experiments of randomly generated grid networks are conducted on Windows Server 2012 R2 Standard with 128 GB RAM and 2.20 GHz processor. All the numerical experiments of real-world traffic networks are conducted on a Windows computer with 32 GB RAM and 3.60 GHz processor.

2.5.2 Results of Randomly Generated Grid Networks

We present the computational time results in Section 2.5.2 and solution performance in Section 2.5.2.

Computational Time Comparison

We set the time limit for Gurobi to 7200 seconds. If we do not apply any decomposition schemes and directly solve the problem, Gurobi is not able to provide a feasible solution or even an upper bound of the objective value within the time limit, indicating the importance of applying decomposition algorithms to solve the model. For Benders decomposition algorithm without temporal decomposition, Gurobi is not able to solve the first-stage problem within the time limit, and thus it cannot give a feasible solution. Based on these results, the temporal decomposition is necessary.

We present the computational time results of the three algorithms with the temporal decomposition in Table 2.3, where we vary the network size extensively. In the table, N_{row} and N_{col} represent the number of intersections of each row and column in the grid networks and "Benders", "Benders-Spatial", "Benders-ADMM" represent Benders decomposition algorithm, spatially decentralized Benders algorithm and ADMM-based spatially decentralized Benders algorithm, respectively. In the table, "MP-min", "MP-max", "MP-A", "SP-min", "SP-max", "SP-A" stand for the minimum, maximum and average computational time in seconds of solving first-stage master problems and second-stage subproblems during all the iterations, respectively. If an algorithm is not able to return a feasible solution due to the time limit, we mark the related results as "-" in the table.

The results show that the computational time varies greatly among different iterations. With added cuts, the computational time of the first-stage problem increases drastically. The computational time of second-stage problems varies depending on first-stage solutions. The results also indicate that for every instance, Benders-Spatial and Benders-ADMM perform significantly better than Benders when solving first-stage problems since the size of first-stage problems is reduced by the spatial decomposition. For second-stage problems, Benders-ADMM outperforms Benders and Benders-Spatial since the parallel computing is able to be applied so that each intersection can solve the second-stage problem at the same time. Moreover, for all the algorithms, the computational time increases when the network

Table 2.3: Computational time of different algorithms for solving grid networks with various sizes

N_{row}	N_{col}	Benders					
		MP-min (s)	MP-max (s)	MP-A(s)	SP-min (s)	SP-max(s)	SP-A(s)
4	4	1.18	123.99	54.98	2005.10	6905.05	4632.75
2	8	0.96	76.20	33.78	1687.63	4834.43	3252.43
6	6	1.78	685.31	132.60	17810.42	43607.42	34603.78
10	10	-	-	-	-	-	-
N_{row}	N_{col}	Benders-Spatial					
		MP-min (s)	MP-max (s)	MP-A(s)	SP-min (s)	SP-max(s)	SP-A(s)
4	4	2.17	16.79	8.00	3437.82	3826.72	3637.06
2	8	1.78	13.31	6.92	2800.51	3579.05	3236.11
6	6	4.65	35.64	17.10	20624.07	27505.60	23339.3
10	10	-	-	-	-	-	-
N_{row}	N_{col}	Benders-ADMM					
		MP-min (s)	MP-max (s)	MP-A(s)	SP-min (s)	SP-max(s)	SP-A(s)
4	4	1.73	46.22	18.81	539.44	594.26	571.98
2	8	1.68	33.96	14.71	295.12	519.81	426.21
6	6	4.26	101.51	42.43	1060.78	1685.95	1404.67
10	10	13.73	77.55	45.64	2780.25	3682.51	3231.38

size is larger. For networks with the same number of intersections, the algorithms take less time to solve the model of an asymmetric network than a symmetric one.

Evaluation Results

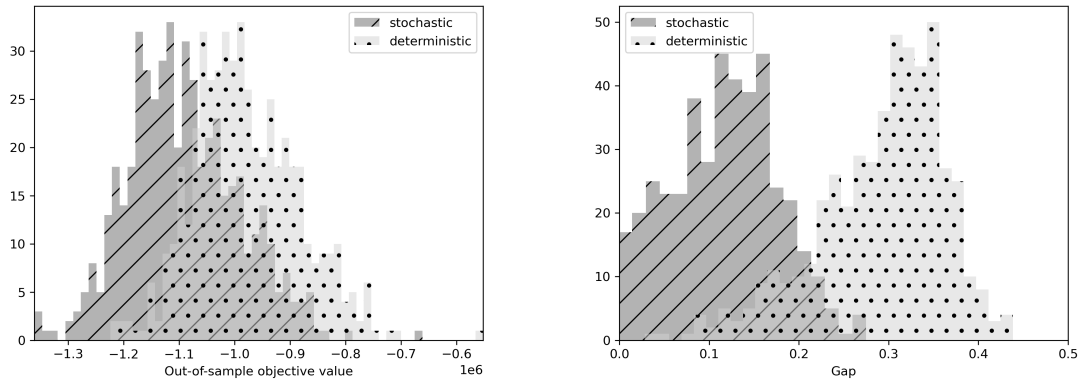
Objective Values We use Benders-ADMM (Algorithm 2) to solve the deterministic (Deter) and stochastic (SP) models and present the objective values of both models in Table 2.4. Column "Mean" presents mean values of traffic demand and Column "SD/Mean" presents the ratios between standard deviations and demand mean values. Columns "In-sample Obj" and "Out-of-sample Obj" present the in-sample and out-of-sample objective values. Column "Gap" presents the gaps between in-sample and out-of-sample objectives. In all cases, gaps of the stochastic model are smaller than gaps of the deterministic model, indicating that the stochastic model describes the real traffic under uncertainties better. When we increase demand mean values, all the objective values increase since there are more vehicles entering the network. When the deviations increase, objective gaps of the deterministic model increase while the stochastic model can still maintain relatively low gaps, showing its solution robustness against parameter uncertainty.

Table 2.4: In-sample and out-of-sample objective values of randomly generated grid networks

Mean (veh/h)	SD/	In-sample Obj (veh·s)		Out-of-sample Obj (veh·s)		Gap	
	Mean	Deter	SP	Deter	SP	Deter	SP
200 (E-W), 50 (S-N)	2	−420597.37	−556225.39	−597997.23	−596944.23	29.65%	6.81%
200 (E-W), 50 (S-N)	3	−420597.37	−649147.73	−715975.68	−727529.79	41.25%	10.76%
200 (E-W), 50 (S-N)	4	−420597.37	−751408.39	−794377.55	−818953.97	47.05%	8.23%
400 (E-W), 100 (S-N)	2	−687354.94	−865289.83	−981940.02	−994091.40	30.00%	12.96%
400 (E-W), 100 (S-N)	3	−687354.94	−986676.06	−978864.21	−1092679.91	29.77%	9.69%
400 (E-W), 100 (S-N)	4	−687354.94	−1044571.92	−1122114.76	−1179086.03	38.74%	9.87%

Figure 2.3 shows the histograms of out-of-sample objective values and the gaps for the instance with 400 veh/h in East-West direction and SD/Mean ratio = 3. Here the rectangles filled with slashes are associated with the stochastic model and the ones with dots are associated with the deterministic model. In most cases, the objective value and the gap of the stochastic model are smaller than the

ones of the deterministic model. The deviation of the gap of the stochastic model is also less than the deterministic model, showing the solution robustness of the former.



(a) Histogram of out-of-sample objective value

(b) Histogram of objective gap

Figure 2.3: Out-of-sample performance and gap results of the instance with demand mean as 400 (E-W), 100 (S-N) and SD/Mean ratio as 3

Evaluation Metrics We select travel delay and throughput as metrics to evaluate signal timing plans. In Table 2.5, we present their values in out-of-sample tests of signal timing plans obtained from the stochastic and deterministic models. Column "Gap" presents the gaps of the corresponding metrics between the deterministic and stochastic model. In most cases, the travel delay of the stochastic model is less than the one of deterministic counterpart and the throughput of the stochastic model is larger than the deterministic counterpart. Therefore, the signal timing plans obtained by the stochastic model outperform the ones of the deterministic model, demonstrating the importance and benefits of considering the uncertainties in traffic networks. When demand mean increases, more improvements are brought by the stochastic model in most cases. For the same mean value, the largest improvement of the stochastic model is often attained when the traffic network is not too idle or too congested (i.e., SD/Mean = 3).

Table 2.5: Out-of-sample evaluation of solutions for randomly generated grid networks with varying demand mean values and SD/Mean ratios

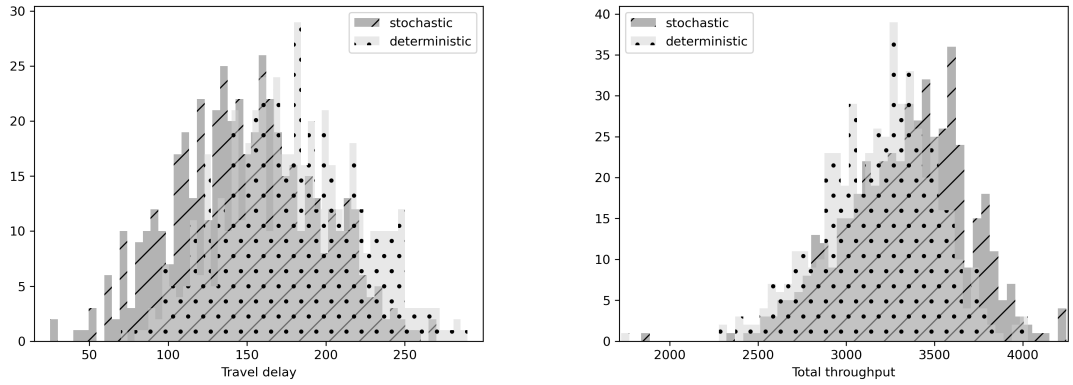
Mean (veh/h)	SD/	Average Travel Delay (s)			Total	Total Throughput (veh)		
	Mean	Deter	SP	Gap	Arrival (veh)	Deter	SP	Gap
200 (E-W), 50 (S-N)	2	29.36	30.00	−2.14%	2105.61	1869.71	1880.02	0.55%
200 (E-W), 50 (S-N)	3	60.20	54.76	9.04%	2872.96	2280.55	2346.54	2.89%
200 (E-W), 50 (S-N)	4	96.81	87.24	8.24%	3658.03	2536.44	2583.79	1.87%
400 (E-W), 100 (S-N)	2	96.67	92.31	4.70%	4195.30	2966.80	3030.21	1.80%
400 (E-W), 100 (S-N)	3	179.12	149.07	16.57%	5825.33	3191.74	3351.85	4.68%
400 (E-W), 100 (S-N)	4	207.95	197.14	5.20%	7300.13	3364.67	3535.66	5.08%

We show the distribution of out-of-sample results in Table 2.6, where we present the standard deviation (SD) and ratio between standard deviation and mean (SD/Mean) of travel delay (D) and throughput (T), indicated by "SD-D", "SD-T", "SD/Mean-D", "SD/Mean-T", respectively. The ratio between standard deviation and mean of travel delay is less than the ratio of the throughput, illustrating that travel delay is more sensitive during the out-of-sample tests. In most cases, when the mean value and deviation of the demand increase, there exist more samples where the traffic network is fully congested, leading to less deviation of traffic delay and throughput. Figure 2.4 shows the histograms of out-of-sample tests on the instance with 400 vehicles per hour in East-West direction and

SD/Mean ratio being 3. In most cases, the stochastic model outperforms the deterministic model in out-of-sample tests.

Table 2.6: Standard deviation of out-of-sample evaluation of solutions for randomly generated grid networks with varying demand mean values and SD/Mean ratios

Mean of Demand (veh/h)	SD/Mean of Demand	SD-D (s)		SD-T (veh)		SD/Mean-D		SD/Mean-T	
		Deter	SP	Deter	SP	Deter	SP	Deter	SP
200 (E-W), 50 (S-N)	2	15.03	14.28	312.95	325.95	0.51	0.48	0.17	0.17
200 (E-W), 50 (S-N)	3	29.75	27.52	334.76	366.29	0.49	0.50	0.15	0.16
200 (E-W), 50 (S-N)	4	37.64	36.83	328.85	308.30	0.39	0.42	0.13	0.12
400 (E-W), 100 (S-N)	2	35.09	34.17	336.78	362.72	0.36	0.37	0.11	0.12
400 (E-W), 100 (S-N)	3	42.18	44.82	318.49	346.43	0.24	0.30	0.10	0.10
400 (E-W), 100 (S-N)	4	48.63	48.11	277.71	325.89	0.23	0.24	0.08	0.09



(a) Histogram of traffic delay

(b) Histogram of throughput

Figure 2.4: Out-of-sample delay and throughput results of the instance with demand mean as 400 (E-W), 100 (S-N) and SD/Mean ratio as 3

2.5.3 Results of Real-world Traffic Networks

In Sections 2.5.3 and 2.5.3, we present the computational time and evaluation results of real-world traffic networks under peak hours and off-peak hours, respectively.

Results of Peak Hours

We present the computational time of the deterministic and stochastic models in seconds in Table 2.7. In the table, "MP-min", "MP-max", "MP-A", "SP-min", "SP-max", "SP-A" are defined the same as before. The table shows that our ADMM-based spatially decentralized Benders algorithm is able to solve both models in acceptable time limit.

Table 2.7: Computational time results of the traffic network of Ann Arbor downtown during peak hours

	MP-min (s)	MP-max (s)	MP-A (s)	SP-min(s)	SP-max (s)	SP-A (s)
Deterministic	0.24	0.79	0.48	41.35	41.66	41.54
Stochastic	1.55	2.67	1.94	447.30	474.69	464.51

We present the in-sample objective values, out-of-sample objective values, and gaps between the deterministic and stochastic models in Table 2.8. To evaluate the traffic timing plans, we also present

the average travel delay, total arrival and total throughput in the second half of the table. In Columns "Average Delay" and "Total Throughput", the percentages in Row "Deterministic" show the improvements of relative metrics compared to the baseline, and the percentages on the row "Stochastic" show the improvements of relative metrics compared to the deterministic model, respectively.

Table 2.8: Out-of-sample evaluation results of Ann Arbor downtown during peak hours

	In-sample Obj (veh·s)	Out-of-sample Obj (veh·s)	Gap
Deterministic	-1498923.97	-1939263.92	22.70%
Stochastic	-1540021.95	-2023592.97	23.88%
	Average Delay (s)	Total Arrival (veh)	Total Throughput (veh)
Baseline	352.76	7956.79	3295.30
Deterministic	297.40 (15.69%)	7956.79	3789.91 (15.01%)
Stochastic	276.33 (7.62%)	7956.79	4009.33 (5.47%)

The results in Table 2.8 show that although the gaps between in-sample and out-of-sample objective values of deterministic and stochastic models are similar, the stochastic model outperforms the deterministic counterpart in terms of the average delay and total throughput. This demonstrates that it is valuable to take into account uncertainties in real-world traffic networks. Comparing the performance of the baseline and deterministic models, the out-of-sample evaluation results of the deterministic model are significantly improved, illustrating the advantages of considering the coordination between different intersections. We visualize the average number of vehicles in the network across all the scenarios over time in Figure 2.5. The line represents the baseline, the line marked by plus signs represents the deterministic model and the line marked by stars represents the stochastic model. We find that the number of vehicles in the traffic network increases fastest in the baseline setting, while it increases slowest in the stochastic model, which also suggests the benefits of stochastic models in preventing congestion. We also provide the visualizations of the number of vehicles under the best and worst scenarios with respect to arrival and delay in Appendix 2.7.

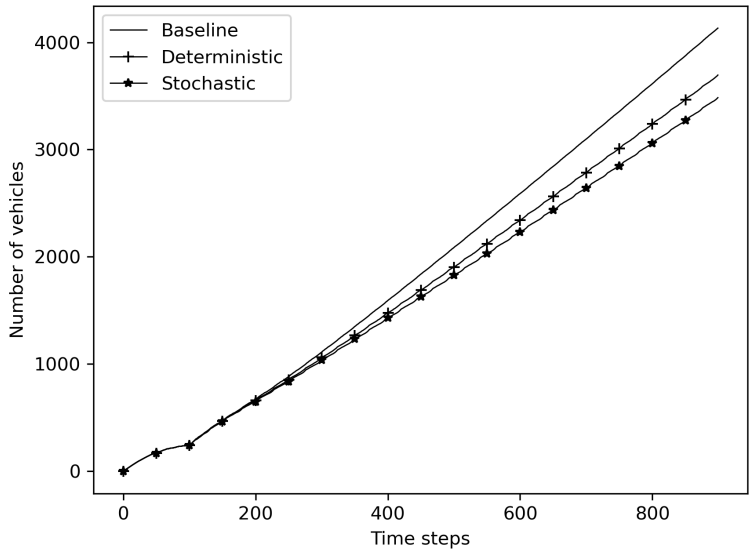


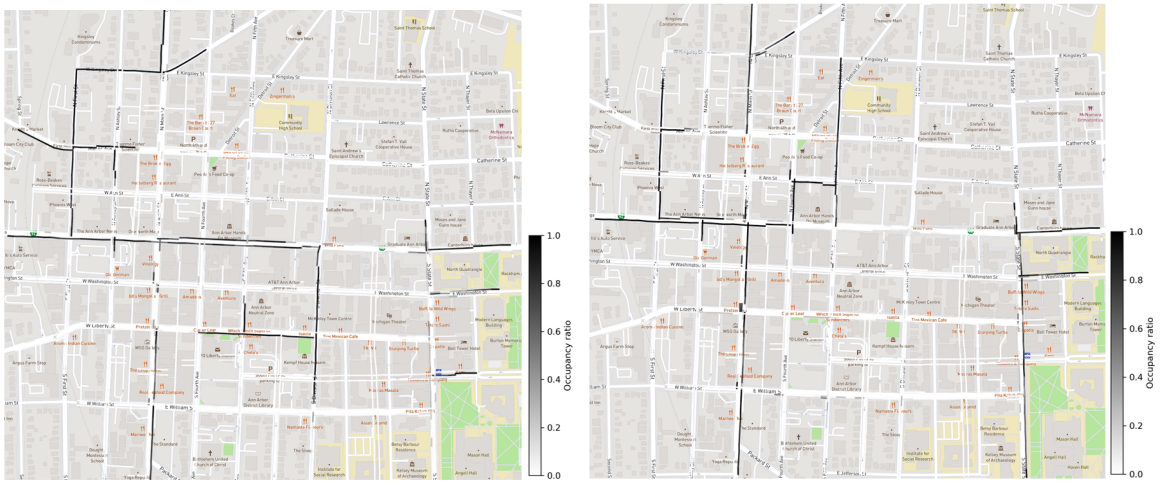
Figure 2.5: Number of vehicles in the traffic network during peak hours with the parameters as mean values

Figure 2.6 provides the snapshots of the spatial distribution of the number of vehicles of deterministic and stochastic models under scenarios with the minimum and maximum delay (i.e., the best scenario and worst scenario) at time step $t = 800$. In the figure, we visualize the occupancy ratio,

which stands for the ratio between the number of vehicles and the maximum allowed number of vehicles of the cell. The road segments are darker if the occupancy ratio is higher, meaning worse traffic congestion. Figures 2.6a and 2.6b show less congestion given by the stochastic model under the best scenario. Comparing Figures 2.6c and 2.6d, although both models have congestion, there are fewer number of congested roads given by the signal plan produced by the stochastic model. The figures also show that for the deterministic model, the congestion in East-West direction is worse while for the stochastic model, the congestion in North-South direction is worse.

(a) Deterministic model under best scenario

(b) Stochastic model under best scenario



(c) Deterministic model under worst scenario

(d) Stochastic model under worst scenario

Figure 2.6: Spatial distribution of vehicles at time step 800 during peak hours

Results of Off-peak Hours

The computational time of the deterministic and stochastic models under off-peak hours are presented in Table 2.9. Both models can be solved within acceptable computational time. Compared to the results under peak hours, it takes less time to solve the models with lower demand.

We compare the performance of the baseline, the deterministic model, and the stochastic model in Table 2.10. We also show the solution improvement of the deterministic model compared to the baseline solution, and the solution improvement of the stochastic model compared to the deterministic one both by percentages. For all the signal timing plans, the average delay of off-peak hours is less than the one of peak hours.

Table 2.9: Computational time results of the traffic network of Ann Arbor downtown during off-peak hours

	MP-min (s)	MP-max (s)	MP-A (s)	SP-min(s)	SP-max (s)	SP-A (s)
Deterministic	0.13	0.40	0.23	40.70	42.36	41.28
Stochastic	0.89	2.72	1.79	424.50	450.26	439.44

Table 2.10: Out-of-sample evaluation results of Ann Arbor downtown during off-peak hours

	In-sample Obj (veh·s)	Out-of-sample Obj (veh·s)	Gap
Deterministic	−580971.48	−1121107.27	48.17%
Stochastic	−592697.54	−1140783.78	48.04%
	Average Delay (s)	Total Arrival (veh)	Total Throughput (veh)
Baseline	117.99	2905.69	2040.82
Deterministic	59.36 (49.69%)	2905.69	2419.07 (18.53%)
Stochastic	51.19 (13.77%)	2905.69	2466.60 (1.96%)

Comparing Table 2.10 with Table 2.8, the gaps between in-sample and out-of-sample objective values of the two models are larger than the gaps during peak hours since the deviation of the traffic demand is higher. The improvements of the deterministic and stochastic models are both more significant in terms of average delay while the improvement of throughput are similar compared to the results of peak hours. We visualize the average number of vehicles in the network across all the scenarios over time in Figure 2.7. The increase rates of the number of vehicles of all models are slower than the results of peak hours since the mean traffic demand of off-peak hours is smaller. In Figure

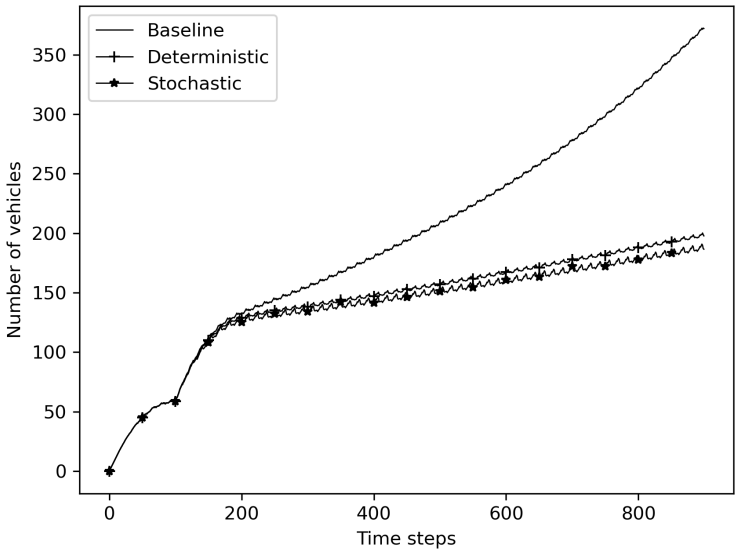
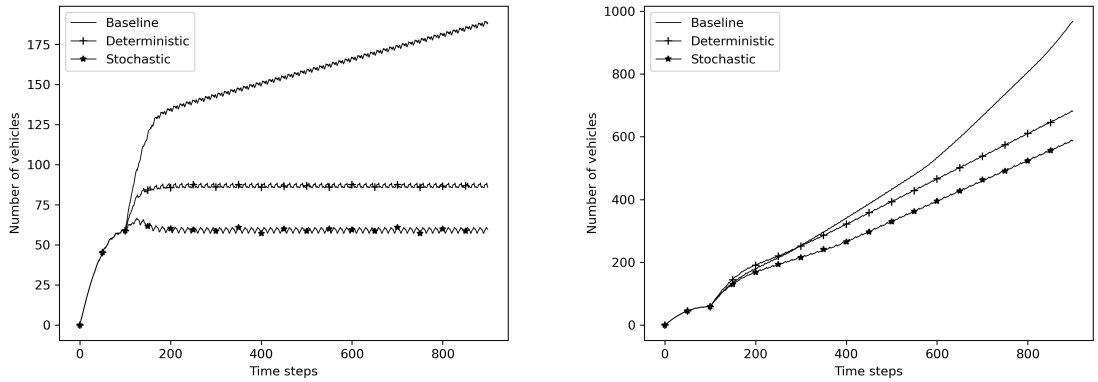


Figure 2.7: Number of vehicles in the traffic network during off-peak hours with the parameters as mean values

2.8, we visualize the number of vehicles in the network of the scenarios with the minimum delay and maximum delay of the deterministic model, stochastic model and baseline setting. Figure 2.8a shows that under the scenario with minimum delay, the number of vehicles in the traffic network keep stable for both deterministic and stochastic models, and there are fewer vehicles of the stochastic model. Figure 2.8b shows that under the scenario with maximum delay, the number of vehicles of all the settings increase and the increase rates of the stochastic model is slower than the other two settings. We also

provide the visualizations of the number of vehicles under the best and worst scenarios with respect to arrival in Appendix 2.7.



(a) Scenario with minimum delay

(b) Scenario with maximum delay

Figure 2.8: Number of vehicles in the traffic network during off-peak hours

Figure 2.9 provides the snapshots of the spatial distribution of the number of vehicles of the deterministic and stochastic models under the best and worst scenario at time step $t = 800$. There are fewer congested intersections in the results of the stochastic model under both scenarios, compared to the peak hours. While the solutions returned by stochastic model can lead to almost fully empty roads, deterministic counterpart still leaves several roads congested.

Based on all the above results, we show the benefits of taking into account the uncertainties as well as the coordination in real-world traffic networks. Our models work well for real-world instances during both peak hours and off-peak hours and our stochastic model is more appropriate for the instances with high standard deviations of travel demand.

2.6 Conclusion

In this paper, we built an MIP for the traffic signal control problem in urban traffic networks based on the CTM. We extended the deterministic model to a two-stage stochastic model considering the uncertainties of traffic demand and turning ratios. We proposed efficient algorithms for solving the models and overcoming the scalability difficulties. Our algorithm not only reduced the computational time but also ensured the optimality for the non-convex model with mixed-integer variables.

With the numerical results obtained from randomly generate grid networks and real-world traffic networks, we firstly showed the reduction of the computational time of our algorithm. Then we demonstrated the benefits of considering the uncertainties in traffic networks. Furthermore, we illustrated the advantages of our model to consider the coordination of all the intersections in a real-world traffic network. We noticed that the parameters in the model need fine-tuning in practice, and this can be achieved since the experiment is offline.

There are several possible directions for future research. Improvements to the distributed algorithms could be applied in the traffic signal control problem to accelerate the computational process. Since our algorithm also works for areas with multiple intersections, it is a valuable topic to group intersections in one area based on the traffic flow data. In addition, with the development of connected and autonomous vehicles (CAV), the distribution of traffic demand and turning ratios can be estimated from real-world data. The combination of data collected from CAV and our proposed approach provides a complete framework for traffic signal optimization in practice.

(a) Deterministic model under best scenario

(b) Stochastic model under best scenario

(c) Deterministic model under worst scenario

(d) Stochastic model under worst scenario

Figure 2.9: Spatial distribution of vehicles at time step 800 during off-peak hours

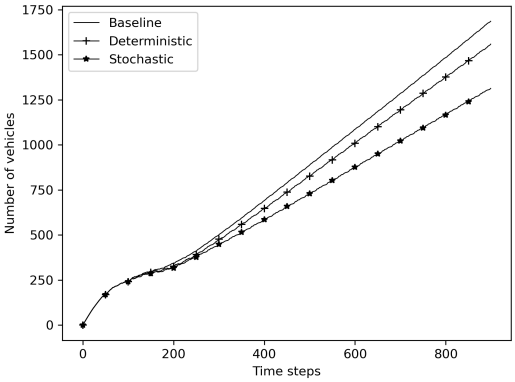
2.7 Appendix

2.7.1 Number of Vehicles in Real-world Traffic Networks

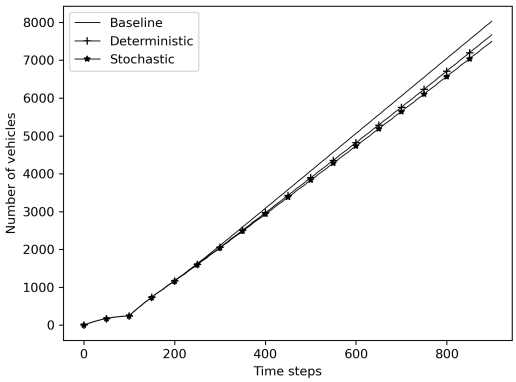
Results of Peak Hours To evaluate the traffic signal plans, in figure 2.10, we visualize the number of vehicles in the network of the scenarios with least and most arrivals. The figure shows that in both scenarios, the stochastic model outperforms than the deterministic model and the baseline, and it is more significant when the arrival is lower.

In Figure 2.11, we visualize the number of vehicles in the traffic network under the scenarios with minimum delay and maximum delay. For both the scenarios, the number of vehicles keep increasing. The increase rates of the stochastic model is lower under the scenario with minimum delay while the increase rate of the deterministic model is lower under the scenario with maximum delay. The increase rate of the stochastic model is higher than the deterministic model because the total arrival of the stochastic model is larger.

Results of Off-peak Hours In figure 2.12, we visualize the number of vehicles in the network of the scenarios with least and most arrivals. Figure 2.12a shows that when the arrival is low, the number of vehicles in the traffic work keep stable for both deterministic and stochastic models. Figure 2.12b shows that when the arrival is high, the increase rates of the number of vehicles of the stochastic model

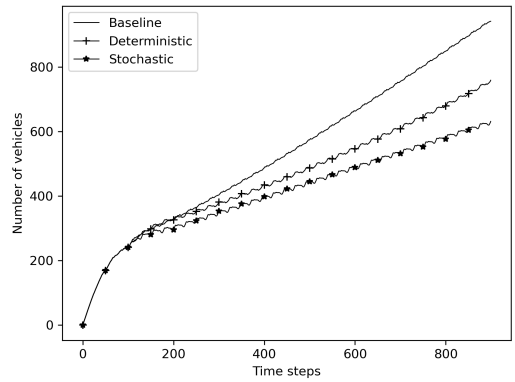


(a) Scenario with lowest arrival

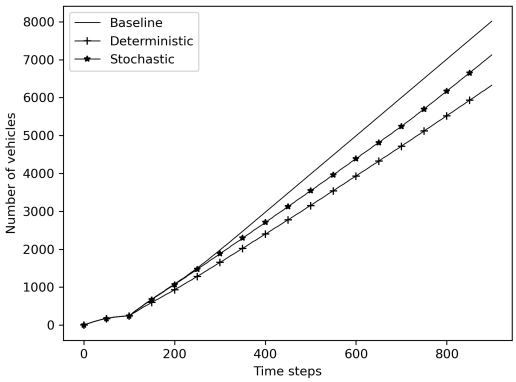


(b) Scenario with highest arrival

Figure 2.10: Number of vehicles in the traffic network during peak hours



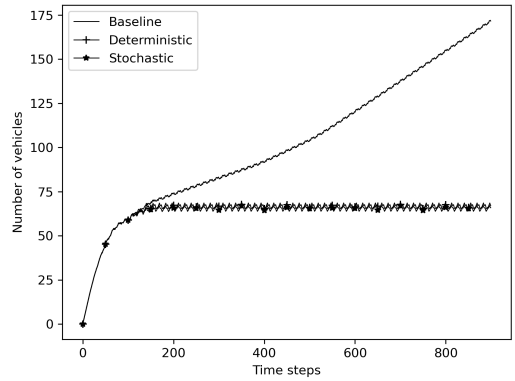
(a) Scenario with minimum delay



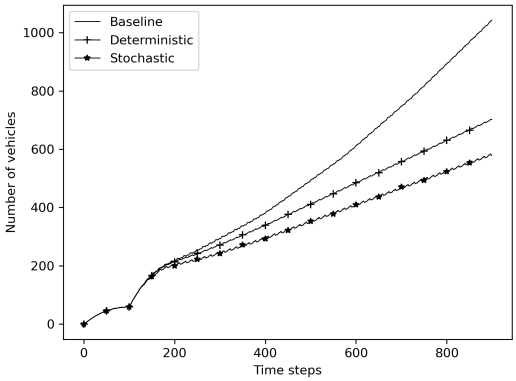
(b) Scenario with maximum delay

Figure 2.11: Number of vehicles in the traffic network during peak hours

is slower than the other two settings.



(a) Scenario with lowest arrival



(b) Scenario with highest arrival

Figure 2.12: Number of vehicles in the traffic network during off-peak hours

Chapter 3

Learning the Max Pressure Control for Urban Traffic Networks

3.1 Introduction

Traffic signal control and optimization methods have been an active research topic for the past decades and recent literature can be roughly divided into three different categories: 1) optimization or optimal control methods based on different traffic models and formulations, 2) artificial intelligence algorithms such as the reinforcement learning (RL), and 3) max pressure control for a general signalized network. Although optimization-based methods have been extensively studied in the past, in recent years, RL-based methods and max pressure control have received significant attention.

Reinforcement learning (RL) algorithms have also been extensively used for traffic signal control optimization during the past decade (Arel et al., 2010; Khamis and Gomaa, 2014; Yau et al., 2017; Chu et al., 2019; Wei et al., 2019c). By training offline, RL can directly learn an end-to-end control policy from the observation by interacting with the simulation environment. Most of the existing literature using RL for traffic signal control focused on the design of the input state space and reward (Wei et al., 2019c), while utilizing different RL techniques such as the multi-agent algorithms (Chu et al., 2019). However, the control policy obtained by RL is usually expressed by a neural network. Due to the issue of the generalization ability of the neural networks, it would not be preferable to directly apply RL policy learned offline in a simulation environment to the real world without additional adjustments.

The max pressure control, which is also known as the back pressure or max weight control, is originally studied in the communication network domain with respect to routing and scheduling (Tassiulas and Ephremides, 1990; Neely, 2010; Srikant and Ying, 2013). It was firstly introduced to traffic network signal control by Varaiya (2013), and followed by various extensions and evaluations (Lioris et al., 2014; Xiao et al., 2014; Le et al., 2015; Zaidi et al., 2016; Sun and Yin, 2018; Manolis et al., 2018; Li and Jabari, 2019; Chen et al., 2020). The max pressure control for urban traffic networks has drawn tremendous attention in recent years since it can provide appealing theoretical guarantee of stabilizing the store-and-forward network as long as the demand is within the network capacity. Besides, it is a decentralized control policy in which each intersection makes its own decision based on the upstream and downstream queue lengths.

However, the max pressure control introduced in most literature (Varaiya, 2013; Le et al., 2015; Xiao et al., 2014; Zaidi et al., 2016) is derived based on a store-and-forward network model, which contains some strong assumptions such as infinite link capacity, no link travel time, and no switching loss. In particular, it is well known that for most of the intersections, due to the phase switching loss, the phase switching frequency should decrease (i.e., cycle length should increase) with the increase of the traffic demand so that the network queue lengths can be stabilized under higher traffic volume without suffering much phase switching loss (Koonce and Rodegerdts, 2008; HCM, 2010). Nonetheless, the conventional max pressure control fails to adjust its phase switching frequency dynamically according to the varying traffic demand; and hence it is no longer throughput-optimal over a network model

with phase switching loss (Celik et al., 2016).

In this work, we propose to utilize the policy-gradient reinforcement learning methods to learn a max pressure control policy that considers the phase switching loss. We first propose an extended max pressure control policy named SCMP, short for Switching-Curve-based Max Pressure control. It can be proved that, under the network model with the phase switching loss, SCMP is *throughput-optimal*, meaning that it can stabilize the network queue lengths as long as the traffic demand is (strictly) within the network capacity. SCMP extends the original max pressure control by introducing a switching curve that could help the controller dynamically adjust the phase switching frequency according to the current traffic loads. To adapt to the real-world traffic which is much more complicated than the store-and-forward point-queue model, we further modify SCMP by using a distributed approximation and the position-weighted pressure scheme. This modified max pressure control, which will be referred as ESCMP (Extended-SCMP), is a more practical and general version of SCMP with the variant weight curves and the switching curves. While the switching curve determines the switching behavior of the controller, the weight curve enables the controller to consider the vehicles at different locations differently, so that it could implicitly improve the coordination among intersections.

Furthermore, we utilize the policy-gradient RL algorithms to optimize the two parametric curves in ESCMP including the switching curve and the weight curve. ESCMP where the parametric curves are optimized by policy-gradient RL algorithms is named as LESCMP (Learned-ESCMP). Compared with other RL-based methods utilizing deep neural networks to represent the actor, LESCMP uses the max pressure control policy network, which is interpretable and derived based on a control policy that has certain theoretical guarantee over a simplified network model. One seemingly similar method to LESCMP that combines the RL and max pressure control comes from Wei et al. (2019a), which integrated the "pressure" into the reward function. However, the difference is quite obvious: we directly utilize the max pressure controller as the actor instead of setting the pressure as the reward.

This paper assumes that the real-time traffic density of each cell on every movement is available as the input of the controller. Therefore, real-time estimation of traffic density is required before the implementation of the proposed control method. As a distributed control policy in which each intersection makes its own decision solely based on its upstream and downstream observation, LESCMP would be also of great significance for the real-world implementation, especially in dealing with large-scale traffic networks.

To sum up, the contributions of this paper are twofold:

- We extend the original max pressure control by adding the phase switching loss and propose a Switching-Curve based Max Pressure (SCMP) control with a stability proof.
- We propose a novel framework utilizing the policy gradient RL algorithms to improve the max pressure control by using a max pressure control policy network.

The rest of this paper is organized as follows. Section 3.2 describes the network model that adds the phase switching loss to the original store-and-forward model. Sections 3.3 introduces the proposed SCMP controller and proves that it is throughput-optimal under the network model with the phase switching loss. Section 3.4 introduces the practical implementation of the SCMP (Extended-SCMP) and then shows how to utilize the policy-gradient reinforcement learning method to optimize the ESCMP controller (Learned-ESCMP). Section 3.5 shows the simulation results while Section 3.6 concludes this paper.

3.2 Network model with switching loss

The original max pressure control (Varaiya, 2013) is established based on the store-and-forward model (Aboudolas et al., 2009), which is essentially a point-queue model and does not consider the phase switching loss. In this section, we modify the store-and-forward model to further capture the phase switching loss. Besides the description in the paper, the notations are also summarized in Appendix 3.7.

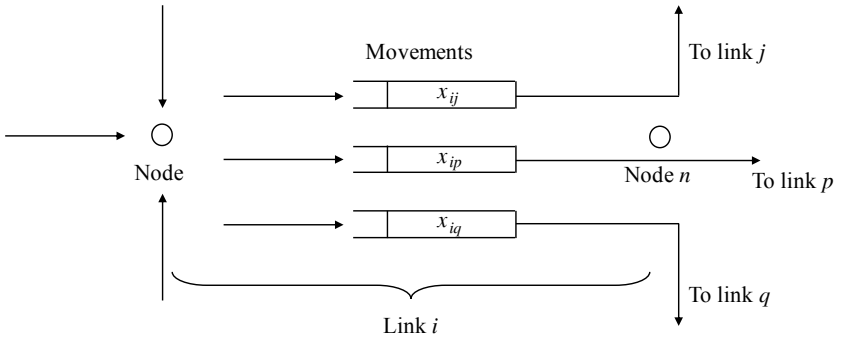


Figure 3.1: Store-and-forward model, reproduced from Varaiya (2013)

Figure 3.1 is an illustration of the store-and-forward network model reproduced from Varaiya (2013). Let $\mathcal{G} = (\mathcal{N}, \mathcal{L})$ represent a general traffic network where $\mathcal{N}$ is the set of the nodes (intersections) and $\mathcal{L}$ represents all the links. A movement can be defined as a tuple composed of the origin link and the destination link $\mathcal{M} = \mathcal{L} \times \mathcal{L}$. Usually, a link contains three movements: through, left-turn, and right-turn. It is assumed that different movements of the same link are separate and do not block each other. We further divide the movements into two categories: ordinary movements $\mathcal{M}_o$ and exit movements $\mathcal{M}_e$. The exit movements will not be considered in the analysis since the vehicle of the exit movements can be freely discharged without additional downstream constraints. For each movement $ij \in \mathcal{M}_o$, let $x_{ij}(t)$ be the queue lengths at time t and c_{ij} be the saturation flow rate, which is treated as a constant.

For the traffic demand and signal constraints, let $a_{ij}(t)$ be the exogenous demand of movement $ij \in \mathcal{M}$, which is assumed to be i.i.d. with the expectation $\mathbb{E}a_{ij}(t) = a_{ij}$ and the maximum value $a_{\max}$ ($0 \leq a_{ij}(t) \leq a_{\max}, \forall t$). Let $r_{ij}(t)$ be the turning ratio from link i to link j, which is also i.i.d. with the expectation $\mathbb{E}r_{ij}(t) = r_{ij}$. For the traffic signal timing plan, we define $s_{ij}(t) \in \{0, 1\}$ as the traffic signal state for the movement ij where 0 corresponds to the red light and 1 represents the green light. Generally, the signal constraints can be formulated as a linear constraint:

$$\boldsymbol{s}(t) \in \{\boldsymbol{s} \mid \boldsymbol{K} \cdot \boldsymbol{s} \leq \boldsymbol{h}\} = \mathcal{S}, \tag{3.1}$$

where $\boldsymbol{s}$ is the column vector that represents the traffic signal state for each movement. $\boldsymbol{K}$ is a matrix and $\boldsymbol{h}$ is a column vector with proper dimensions. For example, if there is only an isolated intersection with two conflict through movements, the signal constraint can be written as $s_1 + s_2 \leq 1$, which can be expressed by Equation (3.1) with $\boldsymbol{s} = [s_1, s_2]^T$, $\boldsymbol{K} = [1, 1]$, and $h = 1$. It is easy to verify that the set $\mathcal{S}$ is a polyhedron with integer-valued vertices. With this traffic signal constraint, a phase is defined as the set of the movements that are allowed to pass the intersection at the same time.

The key intuition of the phase switching loss model is that, for each time when the intersection switches the green time between movements, all the movements in this intersection will have a fixed loss time during which the vehicle is not allowed to pass the intersection. To achieve this, we divide the temporal axis into two different intervals: the discharge interval and the switching interval; each time step of a given movement either belongs to the discharge interval or the switching interval. In the discharge interval, the vehicle in the movement ij is controlled by the traffic signal state s_{ij} and can go to the downstream movement if $s_{ij}(t) = 1$. The switching interval is defined as the period when the intersection is undergoing the phase switching loss, the vehicle cannot pass the intersection no matter what the traffic signal state is. In the real world, the switching interval is composed of part of the yellow time, all the all-red clearance time, and part of the green start.

With the discharge interval and the switching interval, Figure 3.2 is the flowchart shows how we model the phase switching loss and how the movement changes between the switching interval and the discharge interval. For each time step in the discharge interval, the intersection can decide whether to switch the green time. Once it decides to switch the green time, the movement will enter the switching interval; after a fixed loss time T^r, it will recover to the discharge interval. $\chi_{ij}(t)$ is

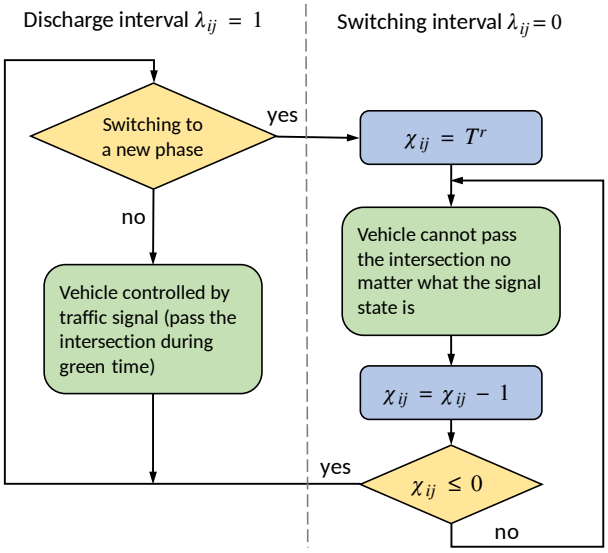


Figure 3.2: Flowchart of the phase switching loss modeling.

the countdown timer that stores the remaining duration of the switching interval and T^r is the total number of time steps of the switching loss.

Let $\lambda_{ij}(t) \in \{0, 1\}$ be the indicator of the intersection interval; $\lambda_{ij}(t) = 1$ means that the movement ij is in the discharge interval while $\lambda_{ij}(t) = 0$ corresponds to the switching interval. $\chi_{ij}(t)$ is the countdown timer of the switching interval, which is 0 in the switching interval while greater than 0 in the discharge interval. To write this with mathematics, we have:

$$\lambda_{ij}(t) = \begin{array}{ll} 1 & \chi_{ij}(t) = 0 \\ 0 & \chi_{ij}(t) > 0 \end{array} \qquad \forall t, \quad \forall ij \in \mathcal{M}_o. \tag{3.2}$$

In the discharge interval, the countdown timer is updated as:

$$\chi_{ij}(t+1) = \begin{array}{ll} T^r & \boldsymbol{s}^n(t+1) \neq \boldsymbol{s}^n(t) \\ 0 & \boldsymbol{s}^n(t+1) = \boldsymbol{s}^n(t) \end{array} \qquad \forall t, \lambda_{ij}(t) = 1 \quad \forall ij \in \mathcal{M}^n_{\text{in}}, \quad \forall n \in \mathcal{N} \tag{3.3}$$

where the superscript n is for the node n; $\mathcal{M}^n_{\text{in}}$ represents the set of the ordinary movements that enter the node n and $\boldsymbol{s}^n$ is the associated signal state vector. Equation (3.3) means that, for every intersection node n, if the traffic signal state $\boldsymbol{s}^n$ changes from t to $t+1$, the countdown timer of all the movements that enter this intersection will be set as T^r; they will enter the switching interval.

When the movement is in the switching interval, the countdown timer is simply updated as:

$$\chi_{ij}(t+1) = \chi_{ij}(t) - 1 \qquad \forall t, \forall ij \in \mathcal{M}_o, \lambda_{ij}(t) = 0. \tag{3.4}$$

To better understand the phase switching loss model given by Equation (3.2-3.4), Figure 3.3 is an example of the signal phase timeline showing how the indicator λ and countdown timer χ are updated. Assuming that we have an isolated intersection with only two conflicted phases (movements) Phase 1 and Phase 2; the traffic signal constraint would be $s_1(t) + s_2(t) \leq 1$ for each time step t. The green and red blocks represent the signal states in the discharge interval while the grey blocks represent the switching interval. The blue dashed lines are the time when the intersection switches green time between the two phases. τ_k is the time step when the kth switching decision is made; after that, both phases enter the switching interval with duration T^r by resetting the countdown timer $\chi = T^r$. As stated before, no matter what the signal state is during the switching interval, vehicles are not allowed to pass the intersection. Therefore, the vehicle can only pass the intersection during the green blocks as shown in the figure. After T^r steps, the countdown timer goes back to 0; the phases recover to the discharge interval and vehicles can pass the intersection during the green time again.

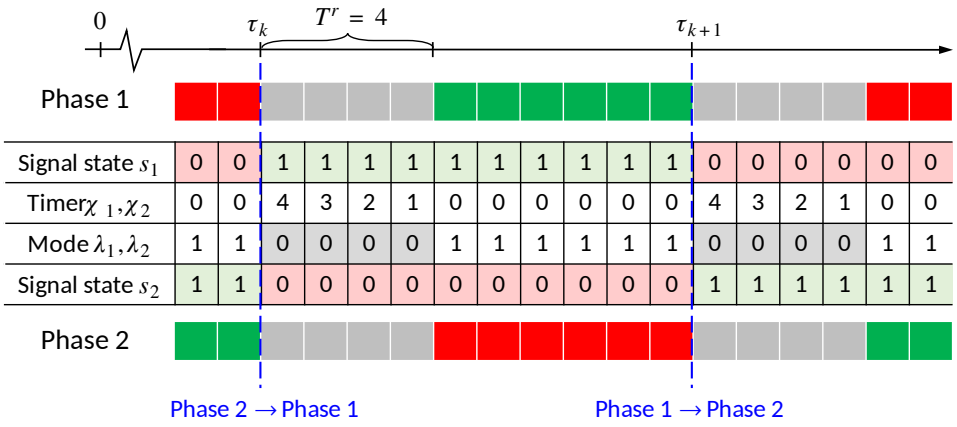


Figure 3.3: Example of the signal phases with switching loss time.

With this phase switching loss model, the signal state s_{ij} alone cannot determine whether the vehicle can pass the intersection, instead, it is determined by the production of the indicator λ_{ij} and the signal state s_{ij}. That is, the vehicle can only pass the intersection when the signal state $s_{ij} = 1$ and the movement is in the discharge interval $\lambda_{ij} = 1$. Therefore, by replacing the signal state $s_{ij}(t)$ with $s_{ij}(t)\lambda_{ij}(t)$ in the original store-and-forward model (Aboudolas et al., 2009; Varaiya, 2013), we will have the dynamics of the queue lengths considering the phase switching loss:

$$\begin{aligned} x_{ij}(t+1) = x_{ij}(t) + a_{ij}(t) + \sum_k r_{ij}(t) \min\{x_{ki}(t), c_{ki}s_{ki}(t)\lambda_{ki}(t)\} \\ - \min\{x_{ij}(t), c_{ij}s_{ij}(t)\lambda_{ij}(t)\} \qquad \forall (i,j) \in \mathcal{M}_o \end{aligned} \tag{3.5}$$

which is equivalent to the matrix form:

$$\boldsymbol{x}(t+1) = \boldsymbol{x}(t) + \boldsymbol{a}(t) - (\boldsymbol{I} - \boldsymbol{R}(t)) \min\{\boldsymbol{x}(t), \boldsymbol{\Lambda}(t)\boldsymbol{C}\boldsymbol{s}(t)\}, \tag{3.6}$$

where $\boldsymbol{x}(t)$ is the queue lengths of all the ordinary movements. $\min\{\cdot,\cdot\}$ is the entry-wise minimization of the two vectors. $\boldsymbol{C}$ is a diagonal matrix with $\boldsymbol{C}_{mm} = c_m, \forall m = (i,j) \in \mathcal{M}_o$. $\boldsymbol{R}(t)$ is the matrix containing all the turning ratio; $\boldsymbol{I}$ is the identical matrix; $\boldsymbol{\Lambda}(t)$ is a diagonal matrix with $\{\boldsymbol{\Lambda}(t)\}_{mm} = \lambda_{mm}(t), \forall m = (i,j) \in \mathcal{M}_o$. Compared with the dynamics in Varaiya (2013), Equations (3.5-3.6) have an extra λ or $\boldsymbol{\Lambda}$ term so that the vehicle is not allowed to pass the intersection in the switching interval.

To sum up, the modified store-and-forward network model that additionally considers the phase switching loss are given by Equation (3.2-3.5). Equation (3.5) is the dynamics of the network queue lengths while Equation (3.2-3.4) shows how indicators that are related to the switching loss model are updated over time.

When there is no switching loss, the whole system is a controlled Markov chain or Markov decision process (MDP) with the system state $\boldsymbol{x}(t)$ and the control policy $\boldsymbol{s}(t)$ for each time step. However, the system is more complicated with the switching loss; the signal state $\boldsymbol{s}(t)$ and the countdown timer $\chi(t)$ need to be augmented to the system state to maintain the Markovian property. Let $S(t) = (\boldsymbol{x}(t), \boldsymbol{s}(t-1), \boldsymbol{\chi}(t))$ be the augmented system state with the dynamics given by Equation (3.2-3.5); it is easy to verify that augmented system state is a controlled Markov chain with system state $S(t)$ and the control policy $\boldsymbol{s}(t)$. The change of the Markovian property will lead to different control policies as well as the stability analysis. Under the store-and-forward network without the switching loss, the control policy (e.g., the max pressure control in Varaiya (2013)) is usually only dependent on the network state $\boldsymbol{x}(t)$. Now with the phase switching loss, intuitively, not only should the control policy be determined by the traffic network state $\boldsymbol{x}(t)$, but also the current signal state $\boldsymbol{s}(t)$.

3.3 Switching-Curved based Max Pressure control and stability analysis

It is easy to verify that, the original max pressure control (Varaiya, 2013) is no longer a throughput-optimal policy based on the modified store-and-forward model with extra consideration of the phase switching loss. In this section, we will introduce the Switching-Curve-based Max Pressure control (SCMP) and proves it as a throughput-optimal policy, which is defined as the control policy that can stabilize the network queue lengths as long as the traffic demand is strictly within the network capacity.

All the analysis in this section is based on the traffic model introduced in Section 3.2; we will also assume that the traffic signal controller have the complete observation and measurement of the network state, i.e., the queue lengths for all the movements before it decides the traffic signal state.

3.3.1 Switching-Curve-based Max Pressure control (SCMP)

The original max pressure control proposed by Varaiya (2013) does not impose any constraints on the switching behavior or the switching frequency; and hence it is not a throughput-optimal policy based on the modified store-and-forward network with the phase switching loss. To address this problem, one of the methods is to design a *hysteresis switching* control (Liberzon, 2003; Lioris et al., 2014), which will tend to keep the signal state unchanged instead of changing the signal state frequently without additional constraints. Specifically, instead of setting the traffic signal state that maximizes the network pressure for each time step, the controller should tend to use the previous traffic signal state unless some additional conditions are satisfied.

The proposed SCMP is inspired by the switching-curve-based (SCB) method introduced in Celik et al. (2016) but is further extended in the two following aspects: 1) from one intersection (single-hop) to a general network (multi-hop); 2) the weight/pressure will be a more general function of queue lengths. Based on the hysteresis switching intuition introduced before, SCMP includes two parts: 1) when to change the traffic signal state, and 2) how to decide the new signal state. For the second part, whenever the switching is activated, the new signal timing plan is chosen as:

$$\boldsymbol{s}^* = \arg\max_{\boldsymbol{s}\in\mathcal{S}} \mathrm{pr}\left(\boldsymbol{x}\left(t\right), \boldsymbol{s}\right) = \arg\max_{\boldsymbol{s}\in\mathcal{S}} \boldsymbol{w}\left(\boldsymbol{x}\left(t\right)\right)^T \boldsymbol{C}(\boldsymbol{I}-\boldsymbol{R})\boldsymbol{s} \tag{3.7a}$$

where $\mathrm{pr}(\boldsymbol{x}, \boldsymbol{s})$ represents the network *pressure* function under the queue lengths vector $\boldsymbol{x}$ and control policy $\boldsymbol{s}$. $\mathcal{M}^n_{\text{in}}$ is the set of upstream movements that enter the node n while $\mathcal{M}^n_{\text{out}}$ is the set of downstream movements that start from the node n. $\boldsymbol{w}(\cdot)$ is to apply function $w_{ij}(\cdot)$ to each entry of the column vector; $w_{ij}(t), \forall ij \in \mathcal{M}_o$ is a weight function that satisfies the following conditions:

1. Function $w_{ij}(x)$ is increasing and continuous for $x \geq 0$, and $w_{ij}(0) = 0$;
2. $w_{ij}(x) \to \infty$ when $x \to \infty$;
3. For any x and Δx, there exists bounded constants $0 < B_0 \leq B_1 < \infty$, such that:

$$w_{ij}(x) + B_0\Delta x \leq w_{ij}(x+\Delta x) \leq w_{ij}(x) + B_1\Delta x \tag{3.8}$$

For example, $w_{ij}(x)$ can be any sublinear or piece-wise linear functions (with a bounded slope) that monotonically increase with the queue lengths. Noted that Equation (3.7a) is essentially a linear program (LP) that will reach the global optimum at an integer-valued vertex. This naturally leads to a binary signal states for each time slot, which suits the case in practice. Another discussion in terms of Equation (3.7) is that it is a distributed algorithm since both the network pressure function and the signal constraints ($\boldsymbol{s} \in \mathcal{S}$) are separable among intersections. Therefore, finding the control policy of the network according to Equation (3.7) is equivalent to find the max pressure control policy for each intersection:

$$\boldsymbol{s}^{n*} = \arg\max_{\boldsymbol{s}^{n*}\in\mathcal{S}^n} \sum_{ij\in\mathcal{M}^n_{\text{in}}} s_{ij}c_{ij}\cdot\left(w_{ij}(x_{ij}) - \sum_{jk\in\mathcal{M}^n_{\text{out}}} r_{jk}w_{jk}(x_{jk}\right) \qquad \forall n \in \mathcal{N} \tag{3.9}$$

where each intersection n can determine its control policy $\boldsymbol{s}^{n*}$ solely based on its upstream observation $\{x_{ij} \mid \forall ij \in \mathcal{M}^n_{\text{in}}\}$ and downstream observation $\{x_{jk} \mid \forall jk \in \mathcal{M}^n_{\text{out}}\}$. Therefore, the max pressure control policy given by Equation (3.7) or Equation (3.9) has two advantages: i) it is a distributed control policy among intersections; ii) it is an end-to-end control policy that directly generates the control policy given the upstream and downstream observation (Varaiya, 2013). The max pressure policy given by Equation (3.9) is similar to the max pressure control given by Varaiya (2013). The only difference is that we generalize the queue lengths in the pressure to a more general function.

With the max pressure signal state given by Equation (3.7) or Equation (3.9). The following is the definition of the *pressure-based control*:

Definition 1. *A control policy is called* ***pressure-based control*** *if it always chooses the max pressure signal state according to Equation (3.7) or Equation (3.9) whenever it decides to change its signal state, otherwise it keeps the signal state unchanged.*

The pressure-based control essentially refers to the controller that always chooses the max pressure phases whenever the controller decides to change the signal phases. Usually, a complete pressure-based control also needs another part about when to switch the signal states. For the original max pressure control in Varaiya (2013), which belongs to the pressure-based control obviously, uses the max pressure signal phases for every time step. Our proposed SCMP also belongs to this pressure-based control; however, instead of setting the signal state as $\boldsymbol{s}^*$ that maximizes the network pressure for every time step, SCMP will keep the signal state $\boldsymbol{s}(t)$ unchanged until a certain switching condition is satisfied. For the switching condition of SCMP, we refer to Celik et al. (2016) and define the switching function as:

$$\begin{aligned}\psi(t) &= \max_{\boldsymbol{s}\in\mathcal{S}} \text{pr}\left(\boldsymbol{x}\left(t\right), \boldsymbol{s}\right) - \text{pr}\left(\boldsymbol{x}\left(t\right), \boldsymbol{s}(t-1)\right) - F\left(\|\boldsymbol{x}\left(t\right)\|\right) \\ &= \max_{\boldsymbol{s}\in\mathcal{S}} \boldsymbol{w}\left(\boldsymbol{x}\left(t\right)\right)^T \boldsymbol{C}(\boldsymbol{I}-\boldsymbol{R})\left(\boldsymbol{s}-\boldsymbol{s}(t-1)\right) - F\left(\|\boldsymbol{x}\left(t\right)\|\right),\end{aligned} \tag{3.10}$$

where $\|\cdot\|$ is 1-norm of the column vector that equals the summation of all the queue lengths. $F(\cdot)$, which is defined as the *switching curve* in this paper, is a monotonically increasing sublinear function satisfying:

$$\lim_{x\to\infty} F(x) = \infty \qquad \lim_{x\to\infty} \frac{F(x)}{x} = 0. \tag{3.11}$$

Based on this switching curve function, the switching is activated if and only if when $\psi(t) \geq 0$, that is,

$$\boldsymbol{s}(t) = \begin{array}{cc} \boldsymbol{s}(t-1) & \psi(t) < 0 \\ \boldsymbol{s}^* & \psi(t) \geq 0 \end{array}. \tag{3.12}$$

To sum up, Figure 3.4 shows the flowchart of the overall SCMP controller given by Equation (3.7-3.12). For each time step, the controller first observes the current system state $\boldsymbol{x}\left(t\right)$ and then finds the timing plan $\boldsymbol{s}^*$ that maximizes the network pressure function. Instead of switching to the new timing plan $\boldsymbol{s}^*$ for every time step, it is only activated whenever the maximized pressure $\text{pr}(\boldsymbol{x}\left(t\right), \boldsymbol{s}^*)$ exceeds the pressure of the original signal timing plan $\text{pr}(\boldsymbol{x}\left(t\right), \boldsymbol{s}(t-1))$ by a certain value $F\left(\|\boldsymbol{x}\left(t\right)\|\right)$, which is a sublinear function of the total queue lengths. With the switching rule given by Equation (3.10-3.12), the switching frequency will decrease with the increase of the traffic demand, since a higher traffic demand usually leads to larger network queue lengths, making the switching condition harder to be satisfied.

Remark 4. *Notice that the switching rule given by Equation (3.10-3.12) is centralized that requires the global queue lengths of the network. This is for the convenience of the stability proof. Although the switching decision is made in a centralized fashion, this does not mean that we require all the intersections to switch at the same time. Whenever the switching is activated, each intersection decides their own traffic signal state according to Equation (3.9); some of the intersections might not change their signal state if the original signal state still maximizes the pressure. But at least one of the intersections will change the signal state otherwise the switching condition*

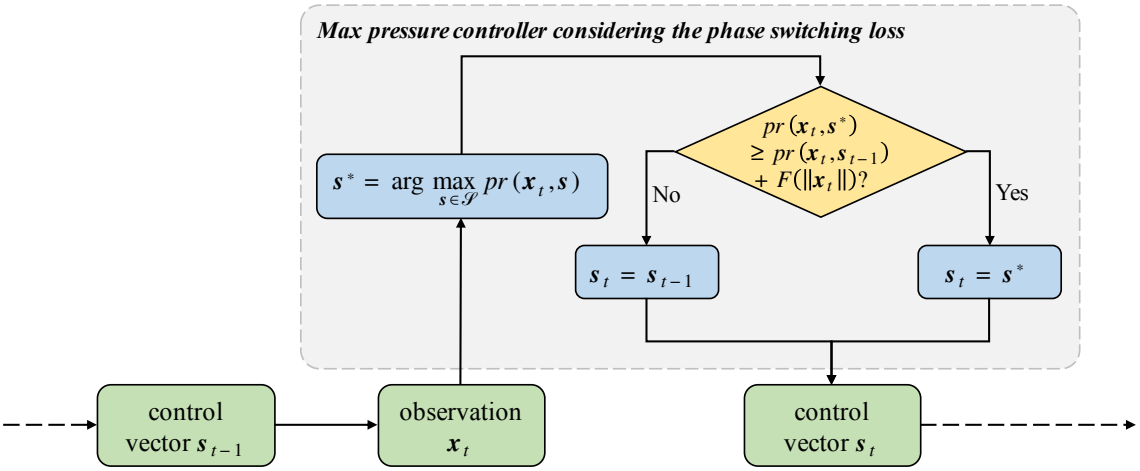


Figure 3.4: Flowchart of the Switching-Curve based Max Pressure control (SCMP).

will not be activated. In Section 3.4, we will use a distributed approximation to make this switching rule also distributed; the approximation is mainly designed for implementation and we will not provide the rigorous proof of the stability.

In the following subsections, we will prove that SCMP is throughput-optimal as long as the weight function and the switching curve satisfy the corresponding conditions.

3.3.2 Network queue lengths stability

Before we go to the detailed proof, we will introduce some preliminary concepts of the network queue length stability. The strong stability of the network queue lengths is defined as (Neely, 2010):

Definition 2. *The network queue lengths are **strongly stable** if:*

$$\lim_{T\to\infty} \sup \frac{1}{T} \mathbb{E} \sum_{t=1}^{T} \|\boldsymbol{x}(t)\| < \infty \tag{3.13}$$

By definition, the strong stability means that average total queue lengths are bounded in the infinite horizon, which indicates that all the demand will be served in the long run.

In Section 3.2, we have defined the signal constraints given by Equation (3.1). The feasible polyhedron of the signal state $\mathcal{S}$ determines the admissible demand region defined below:

Definition 3. *The **admissible demand region** $\mathcal{D}$ is defined as:*

$$\mathcal{D} = \{(\boldsymbol{a}, \boldsymbol{R}) \mid \boldsymbol{a} \preceq (\boldsymbol{I} - \boldsymbol{R})\boldsymbol{C}\boldsymbol{s}, \exists \boldsymbol{s} \in \mathcal{S}\}. \tag{3.14}$$

The admissible demand region defines the feasible average exogenous demand and turning ratio pair $(\boldsymbol{a}, \boldsymbol{R})$ that can be served by the network. Based on this definition of the admissible demand region, a control policy is called *throughput-optimal* if it can stabilize the network queue lengths as long as the demand belongs to the interior of the admissible demand region $(\boldsymbol{a}, \boldsymbol{R}) \in \text{int}\mathcal{D}$. The following theorem shows that, with the phase switching loss, only if the demand is strictly within the admissible demand region, the network queue lengths can be stabilized.

Theorem 8. *Under the network model in Section 3.2 with the switching loss, the network queue lengths can be stabilized **only if** $(\boldsymbol{a}, \boldsymbol{R}) \in int\mathcal{D}$.*

The proof of Theorem 8 can be seen in Appendix Wang et al. (2022). This theorem shows the necessary condition for the network stability: the demand and turning ratio pair has to be strictly within the admissible demand region. This means that no controller can stabilize the network queue lengths if $(\boldsymbol{a}, \boldsymbol{R}) \notin \text{int}\mathcal{D}$.

3.3.3 Sufficient condition for the queue lengths stability with switching loss

With the preliminary concepts introduced in the previous subsection, we will first provide a sufficient condition for the network queue length stability under the network model in Section 3.2. This sufficient condition is modified from Celik et al. (2016), which is extended in two aspects: 1) from a single-hop to a multi-hop network; 2) from the quadratic Lyapunov function to a more general Lyapunov function (Srikant and Ying, 2013).

Let τ_k be the time step when the kth switching is activated and $\boldsymbol{s}_k$ be the signal timing plan chosen after kth switching. In this section, we redefine the augmented system state as $\tilde{S}(t) = (k, t, \tau_k, \boldsymbol{s}_k, \boldsymbol{x}(t))$, which indicates that, by time t, the traffic signal has switched k times and the kth switching changed the traffic signal state to $\boldsymbol{s}_k$ at time τ_k. According to this definition, $(t - \tau_k)$ is the elapsed time of the latest traffic signal $\boldsymbol{s}_k$; the countdown timer $\chi_{ij}(t)$ defined in Section 3.2 is determined by:

$$\chi_{ij}(t) = \begin{array}{cc} \max\left(t - \tau_k - T^r\right), 0\} & ij \in \mathcal{M}^n_{\text{in}}, \boldsymbol{s}^n_{k-1} \neq \boldsymbol{s}^n_k \\ 0 & ij \in \mathcal{M}^n_{\text{in}}, \boldsymbol{s}^n_{k-1} = \boldsymbol{s}^n_k \end{array} \quad \forall t, \forall n \in \mathcal{N} \tag{3.15}$$

It is easy to verify that, under this new augment system state $\tilde{S}(t)$ and the dynamics given by Equations (3.2-3.5,3.15), the system is still a controlled Markov chain.

Under the weight function $w(\cdot)$, the Lyapunov function of a given network queue length state $\boldsymbol{x}\,(t)$ is defined as:

$$L(\boldsymbol{x}\,(t)) = \sum_{ij \in \mathcal{M}^o} \int_{\zeta=0}^{x_{ij}(t)} w_{ij}(\zeta) d\zeta, \tag{3.16}$$

which becomes the quadratic Lyapunov function used in Varaiya (2013) when the weight function $w_{ij}(\zeta) = \zeta$. The following theorem provides a sufficient condition for the network queue length stability under the switching loss.

Theorem 9. *Given a pressure-based control policy defined in Definition 1, for each k, τ_k is the kth switching time. There exists a τ'_{k+1}, which is a random stopping time conditional on system state $\tilde{S}(\tau_k)$ and always between τ_k and τ_{k+1} for all sample paths. If the following conditions are satisfied for each k:*

$$\tau_{k+1} \geq \tau'_{k+1}, \text{ for all sample paths;} \tag{3.17a}$$
$$\mathbb{E}[(\tau'_{k+1} - \tau_k)|\tilde{S}(\tau_k)]. \geq c_1(1 - \delta'(\|\boldsymbol{x}(\tau_k)\|))F(\|\boldsymbol{x}(\tau_k)\|); \tag{3.17b}$$
$$\mathbb{E}[(\tau'_{k+1} - \tau_k)^2|\tilde{S}(\tau_k)]. \leq T_r^2 + c_2(F(\|\boldsymbol{x}(\tau_k)\|))^2; \tag{3.17c}$$
$$\mathbb{E}[L(\boldsymbol{x}(t+1)) - L(\boldsymbol{x}(t))|\tilde{S}(t)]. \leq c_3 - \epsilon\|\boldsymbol{w}(\boldsymbol{x}(t))\|, \quad \forall t \in \{\tau'_{k+1}, \tau'_{k+1} + 1, ..., \tau_{k+1}\}; \tag{3.17d}$$

the network queue lengths will be strongly stable when the demand belongs to the interior of the admissible demand region $(\boldsymbol{a}, \boldsymbol{R}) \in int\mathcal{D}$. In these equations, $F(\cdot)$ is a sublinear function satisfying Equation (3.11); $\delta'(\cdot)$ is a non-negative function with $\lim_{xarrow\infty} \delta'(x) = 0$; ϵ, c_1, c_2, c_3 are bounded positive constants.

Theorem 9 provides a sufficient condition for the network queue lengths stability for the pressure-based control in Definition 1. Recap that the pressure-based control will always choose the max pressure control whenever the switching is activated; Equation (3.17a-3.17d) in Theorem 9 are essentially some additional requirements for the switching condition.

The temporal axis illustration given by Figure 3.5 could help to understand this theorem. τ'_{k+1} is an intermediate time between the kth switching and the $(k+1)$th switching. The second and the third condition given by (3.17b-3.17c) restrict the expectation of the first and second order of the temporal difference $\tau'_{k+1} - \tau_k$. For the first condition given by (3.17a), if this condition holds as equality, then the last condition will be redundant. If it holds as inequality instead, which means that the $(k+1)$th switching is activated after τ'_{k+1}, then we need an extra condition given by Equation (3.17d), requiring an upper bound for the step-by-step Lyapunov drift from τ'_{k+1} to τ_{k+1}.

With Jensen's inequality, from Equation (3.17c) we have:

$$\mathbb{E}[\tau'_{k+1} - \tau_k|\tilde{S}(\tau_k)]. \leq T_r + \sqrt{c_2}F(\|\boldsymbol{x}(\tau_k)\|) \tag{3.18}$$

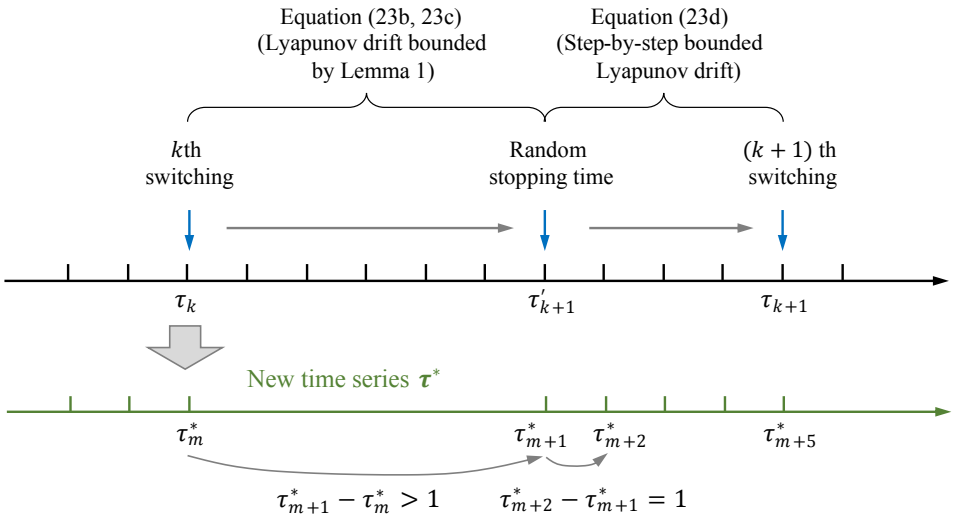


Figure 3.5: Temporal axis of the switching times.

and thus we have the double-sided bound for expectation of the temporal difference between the τ_k and the random stopping time τ'_{k+1} given by Equation (3.18) and Equation (3.17b).

The proof of this theorem is based on a lemma that bounds the Lyapunov drift from the switching time τ_k and the random stopping time τ'_{k+1} for each k. Let Δ'_{τ_k} be the conditional Lyapunov drift from τ_k to the random stopping time τ'_{k+1}:

$$\Delta'_{\tau_k} = \mathbb{E}[L(\boldsymbol{x}(\tau'_{k+1})) - L(\boldsymbol{x}(\tau_k))|\tilde{S}(\tau_k)], \tag{3.19}$$

then the lemma can be written as:

Lemma 3. *If the pressure-based control satisfies the conditions given in Theorem 9, then given the demand within the admissible demand region, there exists $\eta > 0, c_4 < \infty$, such that:*

$$\Delta'_{\tau_k} \leq c_4 - \eta F\left(\|\boldsymbol{x}\left(\tau_k\right)\|\right)\|\boldsymbol{w}\left(\boldsymbol{x}\left(\tau_k\right)\right)\| \tag{3.20}$$

The proof of Lemma 3 is in Wang et al. (2022). Lemma 3 essentially provides an upper bound for the Lyapunov drift from the time τ_k to τ'_{k+1}. With Lemma 3, here we give a sketch of the proof of Theorem 9.

Proof of Theorem 9 As shown on the green axis in Figure 3.5, let $\boldsymbol{\tau}^*$ be a new time series that skips the time slots when $\tau \in (\tau_k, \tau'_{k+1}), \forall k$, that is,

$$\boldsymbol{\tau}^* = [\tau_1^*, \tau_2^*, \tau_3^*, ...]^T = \left[\tau_0, \tau'_1, \tau'_1 + 1, ..., \tau_1, \tau'_2, \tau'_2 + 1, ..., \tau_2, ...\right]^T. \tag{3.21}$$

We will first show that the global queue lengths are strongly stable in this new time series $\boldsymbol{\tau}^*$ and then complete the proof by extending the results to the whole time series. It is easy to verify that under the new time series, the augmented system state $\tilde{S}(\tau_i^*)$ is still a Markov chain. Combining the Lemma 3 and the last condition in Theorem 9, we have:

$$\mathbb{E}[L(\boldsymbol{x}(\tau_{i+1}^*)) - L(\boldsymbol{x}(\tau_i^*))|\tilde{S}(\tau_i^*).] \leq c' - \epsilon\|\boldsymbol{w}(\boldsymbol{x}(\tau_i^*))\|, \quad \forall i. \tag{3.22}$$

where $c', \epsilon > 0$. This equation comes from Lemma 3 when $\tau_{i+1}^* - \tau_i^* > 1$ and from Equation (3.17d) when $\tau_{i+1}^* - \tau_i^* = 1$. Taking the expectation for both sides of this equation and summing up all the equations for all i from 0 to T, we will have:

$$\begin{aligned} &\mathbb{E}\left(L\left(\boldsymbol{x}\left(\tau_T^*\right)\right)\right) - \mathbb{E}(L(\boldsymbol{x}(0))) \leq Tc' - \epsilon\sum_{i=1}^{T}\mathbb{E}\left(\|\boldsymbol{w}\left(\boldsymbol{x}\left(\tau_i^*\right)\right)\|\right) \\ \Longrightarrow \quad &\frac{1}{T}\sum_{i=1}^{T}\mathbb{E}\left(\|\boldsymbol{w}\left(\boldsymbol{x}\left(\tau_i^*\right)\right)\|\right) < \frac{c' + \mathbb{E}(L(\boldsymbol{x}(0)))}{\epsilon} < \infty, \quad \forall T. \end{aligned} \tag{3.23}$$

Since the network queue lengths $\|\boldsymbol{x}(t)\|$ can be bounded by $B \cdot \|\boldsymbol{w}(\boldsymbol{x}(t))\|$ with a bounded positive constant B according to the properties of the weight function given by Equation (3.8); we have:

$$\frac{1}{T}\sum_{i=1}^{T}\mathbb{E}\left(\|\boldsymbol{x}(\tau_i^*)\|\right) < B \cdot \frac{1}{T}\sum_{i=1}^{T}\mathbb{E}\left(\|\boldsymbol{w}(\boldsymbol{x}(\tau_i^*))\|\right) < \infty, \forall T \Longrightarrow \lim_{T\to\infty}\sup\sum_{i=1}^{T}\frac{1}{T}\mathbb{E}\left(\|\boldsymbol{x}(\tau_i^*)\|\right) < \infty, \quad (3.24)$$

which means that the network queue lengths under the new time series $\boldsymbol{\tau}^*$ is strongly stable according to Definition 2. To prove that the queue lengths are strongly stable of the whole time series, the remaining issue is to prove the queue lengths are also bounded for every time steps in the skipped period $(\tau_k, \tau'_{k+1}), \forall k$. Here we choose to omit the detailed proof for this part, which turns out to be easy but tedious. The intuition is that the queue lengths within the skipped time steps can be easily bounded by Equation (3.18) and the bounded arrival assumption ($0 \leq a_{ij}(t) \leq a_{\max}, \forall ij, t$). This omitted part would be similar to the proof of Theorem 1 in Celik et al. (2016), although it focused on a single-hop network (isolated intersection). □

3.3.4 Stability of SCMP

With the sufficient condition of the queue lengths stability given in the previous subsection, this subsection will show that SCMP given by Equation (3.7-3.12) is throughput-optimal which means that it could stabilize the network queue lengths as long as the traffic demand is strictly within the network capacity. Before we prove the stability of the proposed control policy, we will first introduce a lemma which defines a *biased-based* policy (Celik et al., 2016; Hsieh et al., 2017) that can stabilize the network queue lengths by satisfying the sufficient conditions in Theorem 9 with the first condition as an equality.

Lemma 4. *Given a control policy that chooses the max pressure policy according to Equation (3.7) whenever the switching is activated, for each k (pressure-based control), if the switching is activated when the the following $\psi'(\cdot)$ function*

$$\psi'(t) = \|\boldsymbol{x}(t) - \boldsymbol{x}(\tau_k)\| - \theta F\left(\|\boldsymbol{x}(\tau_k)\|\right), \quad \theta > 0 \quad (3.25)$$

is greater than zero, the control policy satisfies the conditions in Theorem 9 with the first condition as equality. Such pressure-based control based on a new switching function $\psi'(\cdot)$ is called biased-based control.

Biased-based policy also belongs to the pressure-based policy in Definition 1. It is similar to SCMP but uses a different switching function $\psi'(\cdot)$. For each switching k, the first term in the $\psi'(\cdot)$ function is the total variation between the current network queue lengths $\boldsymbol{x}(t)$ and the queue lengths at τ_k when the latest switching happens. Therefore, the biased-based control chooses to switch to the max poressure signal state whenever the total variation of the network queue lengths is larger than a certain value. We choose not to go through the details of the proof of this Lemma since it is similar to the proof in Celik et al. (2016). Basically, it can be easily derived by using some relaxations to bound the first- and second-order moments of the temporal difference between the previous switching time τ_k and the first time when the $\psi'(\cdot)$ function given by Equation (3.25) is greater than 0. With Lemma 4, the following theorem shows that, when the demand is strictly within the admissible demand region, SCMP given by Equation (3.7-3.12) satisfies the condition in Theorem 9, and hence can stabilize the network queue lengths.

Theorem 10. *When the demand is strictly within the admissible demand region, SCMP given by Equation (3.7-3.12) satisfies the condition in Theorem 9 with the first condition as inequality, and hence could stabilize the network queue lengths.*

The proof of Theorem 10 is provided in Appendix Wang et al. (2022). The basic idea of the proof is to first show that before the switching is activated according to the switching rule given by Equation (3.10), there is always a corresponding biased-based policy given by Equation (3.25) that is activated in advance. This means that the second and the third condition of Theorem 9 are satisfied by Lemma 4. After that, we can get an upper bound for the Lyapunov drift step-by-step by using the fact that

the condition given by Equation (3.10) is not satisfied yet. This upper bound for the step-by-step Lyapunov drift guarantee the Equation (3.17d) in the sufficient condition is true. With all these together, Theorem 10 eventually shows that SCMP is throughput-optimal under the store-and-forward network model with the phase switching loss by satisfying all the conditions given by Equation (3.17a-3.17d) in Theorem 9.

3.4 Optimizing the max pressure controller using policy-gradient reinforcement learning

In Section 3.2 and 3.3, we add the phase switching loss to the network model and propose a SCMP controller that is proved to be throughput-optimal. Although the theoretical analysis in Section 3.2 and 3.3 additionally considers the phase switching loss which is not included in the original max pressure control, the theoretic SCMP controller might still not suit the real-world traffic very well since the store-and-forward model has some other strong assumptions such as the point-queue assumption.

To further adapt the theoretic SCMP controller to the real world implementation, in this section, we propose a practical version called Extended-SCMP (ESCMP) by using a distributed approximation and also combining some insights from another work from Li and Jabari (2019), which allows us to calculate the movement pressure according to the different location of the vehicle. ESCMP is a practical and heuristic controller based on our previous theoretical analysis; and hence we will not provide the rigorous stability proof as the previous sections. In fact, it would be much more complicated and intractable to provide a theoretical analysis based on a more realistic traffic model. With this, we also propose a novel framework utilizing the policy-gradient RL algorithms to optimize the parameters in the ESCMP controller including the weight curve and the switching curve.

3.4.1 Practical implementation: ESCMP

As an extension for more implementation consideration, ESCMP modifies and extends SCMP in two aspects: 1) from a centralized switching to an approximated distributed switching; 2) from the weight or pressure defined by the function of the queue lengths to a more general position weighted pressure.

Distributed switching As aforementioned in Remark 4, although the selection of $\boldsymbol{s}^*$ given by Equation (3.7) is distributed among intersections, Equations (3.10-3.12) require the switching time to be determined in a centralized fashion. The main reason for choosing a centralized switching rule is to simplify the proof of the global stability. If each intersection decides its own switching time, it would be difficult to analyze the Lyapunov drift of intersections with different switching times. To overcome the similar difficulties, in Hsieh et al. (2017), a superframe is pre-determined by collecting the queue lengths of all the intersections and then individual intersections are allowed to switch more frequently within the superframe. However, it would be better if the switching rule is decentralized which means that each intersection can decide to switch or not only using the local information, making it easier for the real-world implementation. Besides, although the centralized switching is proved to be a stable policy, it has the effect of forcing the intersections with lower traffic volumes to switch less frequently to be synchronized with those congested intersections. This might increase the delay of the low volume intersections.

Therefore, ESCMP uses an approximated distributed switching to replace the centralized switching rule; each signalized node decides to switch whenever the function $\psi^n(\cdot)$ defined below is greater than zero:

$$\psi^n(t) = \max_{\boldsymbol{s}^n \in \mathcal{S}^n} \operatorname{pr}\left(\boldsymbol{x}^n(t), \boldsymbol{s}^n\right) - \operatorname{pr}\left(\boldsymbol{x}^n(t), \boldsymbol{s}^n(t-1)\right) - F\left(\|\boldsymbol{x}^n(t)\|\right) \qquad \forall n \in \mathcal{N}, \tag{3.26}$$

where the superscript n refers to the corresponding value of the node n. Specifically, $\boldsymbol{x}^n$ and $\boldsymbol{s}^n$ represent the queue lengths and traffic signal states of all the movements that enter the node n accordingly. Compared with the switching curve given by Equation (3.10), the switching curve given by Equation (3.26) only requires the number of vehicle in local intersection instead of the number of vehicle in the entire network.

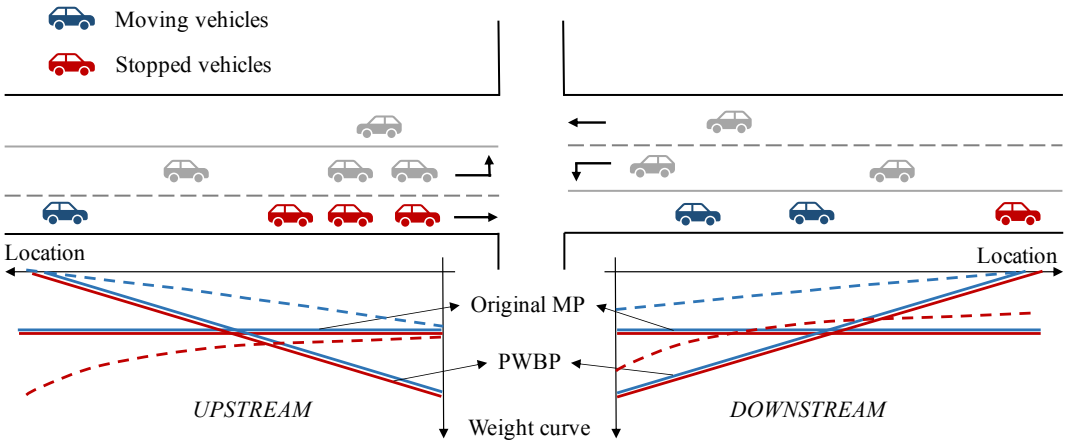


Figure 3.6: Position-weighted curve to calculate the pressure. Original MP: original max pressure proposed by Varaiya (2013), PWBP: position-weighted back pressure proposed by Li and Jabari (2019). As shown by the dashed lines, this paper regards the position-weighted curves as the parameters to be optimized.

Position weighted pressure Similar to other max pressure controllers proposed before (Varaiya, 2013; Le et al., 2015; Zaidi et al., 2016), SCMP is derived based on a store-and-forward point-queue network model. One of the major limitations of the store-and-forward model is that it does not consider the vehicle distribution along the link, nor the spatial propagation. To address this problem, Li and Jabari (2019) proposed a position weighted back pressure (PWBP) control, which used a position weighted method (a linear weight) to calculate the pressure for each movement. We will use the similar idea to get the pressure for each movement. Figure 3.6 illustrates the position-weighted pressure as well as different weight curves. The moving vehicle and stopped vehicle might be considered separately, the red lines are weight curves for the stopped vehicle while the blue lines correspond to the moving vehicle. The key idea of the position weighted pressure proposed by Li and Jabari (2019) is that vehicles at different locations of the movement might contribute differently to the movement pressure. Let $\{l_{ij,p}, \forall p\}$ be the set of the locations (represented by the traveled distance from the start of the movement) where there is a vehicle p along the movement ij; $w^*_{ij}(\cdot)$ is the position-based weight function as shown by the different curves in Figure 3.6. For the position weighted pressure, the $w_{ij}(x_{ij})$ in Equation (3.9) is replaced by:

$$w_{ij}(x_{ij}) \rightarrow \tilde{w}_{ij}(\{l_{ij,p}, \forall p\}) = \sum_p w^*_{ij}(l_{ij,p}) \tag{3.27}$$

To differentiate the $w_{ij}(\cdot)$ function in Section 3.3, we use $\tilde{w}_{ij}(\cdot)$ to represent the weight calculated by the position-weighted method. The weight function $w_{ij}(\cdot)$ in Section 3.3 is a general function of the movement queue lengths x_{ij} while the weight function $\tilde{w}_{ij}(\cdot)$ here is calculated by the weighted sum given the different specific location of the vehicle along the movement $\{l_{ij,p}, \forall p\}$.

Under this position weighted pressure scheme, Li and Jabari (2019) used a linear curve to calculate the movement pressure as shown in Figure 3.6, which means that the closer a vehicle to the intersection, the more it contributes to the movement pressure. As a comparison, the original max pressure control proposed by Varaiya (2013) directly used the queue lengths as the pressure for each movement. Under the store-and-forward model, the queue length is essentially the number of vehicles within the movement. Therefore, the vehicles of the same movement exert equal pressure to the whole movement as shown in the horizontal lines in Figure 3.6. Both the original MP and PWBP do not distinguish the moving vehicle and the stopped vehicle; and hence the blue line and the red line are overlapped. In this paper, as shown by the dashed line, we will treat the weight curve as the parameter to be optimized and split the vehicles to moving vehicles and stopped vehicles. Intuitively, an increasing function would be preferable for moving vehicles. On the contrary, a decreasing function might be better for the stopped vehicles since it can penalize the long queues. Although we do not provide the

theoretical analysis of the stability of this position weighted pressure scheme based on a first-order traffic flow model like Li and Jabari (2019), we do use a general weight function of the queue lengths to get the pressure under the store-and-forward model, trying to mimic the similar effect.

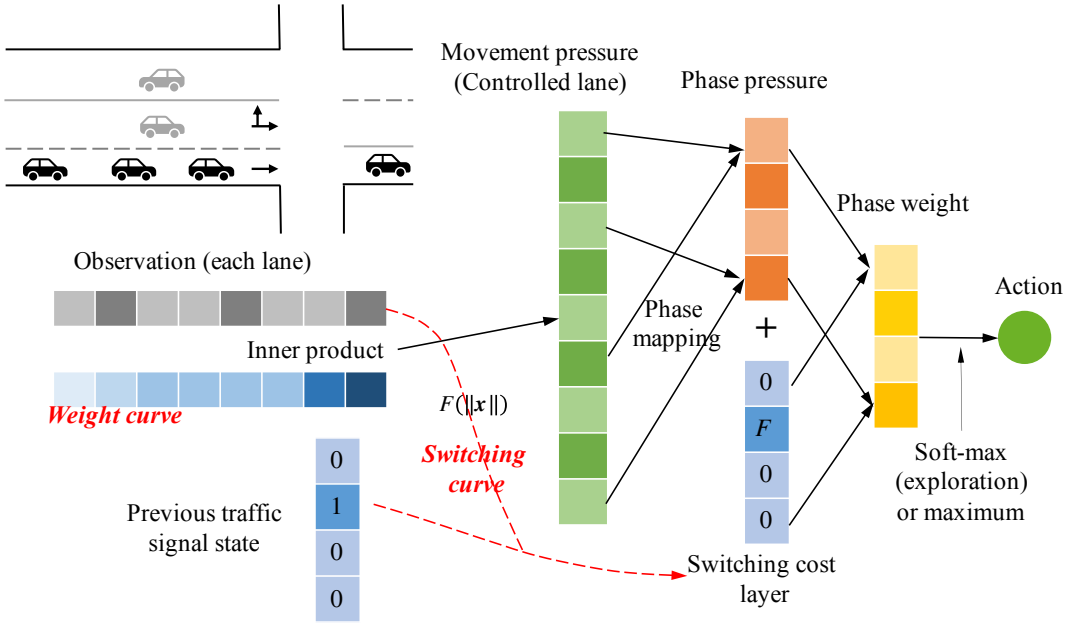


Figure 3.7: The max pressure control policy network (ESCMP).

With these two adaptations, Figure 3.7 is an illustration to the overall ESCMP controller. As a distributed controller, each intersection has a controller of the same structure as shown in the figure but they would have different parameters due to the different geometry and demand patterns. The road segment is discretized into cells with equal length so that the observation is the number of vehicles within each cell. Therefore, ESCMP controller essentially takes the real-time density of each cell on every movement as the input, which would require a real-time traffic state estimation before applying the controller.

With the observation as the number of vehicles in each cell, the weight curve is represented by a vector that has the same dimension as the observation. In other words, instead of using a continuous weight curve as shown in Figure 3.6 and Equation (3.27), we use the same weight for the vehicle in the same cell as a discrete simplification. The movement (corresponds to a controlled lane) pressure is obtained by performing an inner product over the observation and the weight curve. A phase is defined as the set of movements that are allowed to pass at the same time, corresponding to a feasible control policy vector $s^n \in \mathcal{S}^n$ of the node n. Then the phase pressure is the summation of the pressure of the movements that are allowed to pass during this phase. Before getting the phase weight layer, the phase pressure should add another switching cost layer, which determines the switching frequency of the controller. At last, the phase with the maximum weight will be chosen as the action of the current time slot.

3.4.2 Parameter optimization using policy-gradient methods: LESCMP

With the policy network of ESCMP given by Figure 3.7, we are able to leverage the policy-gradient methods to optimize the parameters including the weight curve and the switching curve. Unlike most the literature that used the deep neural networks as the actor in the RL algorithms (Chu et al., 2019; Yau et al., 2017; Khamis and Gomaa, 2014; Arel et al., 2010), we use RL to optimize a policy network that has a pre-determined max pressure structure. In this paper, ESCMP controller that is further optimized by RL is named as Learned-ESCMP (LESCMP).

RL algorithms and the LESCMP controller can further improve the theoretic SCMP controller in two aspects. Firstly, it can take the coordination among intersections into consideration by adjusting

the location-based weight curve when trying to maximize the system total reward. Secondly, the theoretical analysis with regard to the max pressure controller only concerns the stability of the system, which means that the total queue lengths are bounded or the traffic demand can be served in the long run. According to Little's law (Little and Graves, 2008), the bounded total queue lengths guarantee a bounded total delay but not the optimal total delay. Therefore, RL algorithms can be utilized to further optimize the delay performance of the system, which turns out to be hard to deal with in the theoretical analysis.

With the switching loss, the system is still a Markov chain by augmenting the traffic signal state and the countdown timer to the traffic state representation. Let S_t be the augmented system state at time slot t. The weight curves and switching curve in Figure 3.7 can be parameterized by $\theta \in \Theta$. Since RL requires a stochastic control policy to perform the exploration, the deterministic max pressure control policy can be easily converted to a stochastic version by changing the maximization operator to a softmax (logit model) when selecting the final action as shown in Figure 3.7. Let $\pi_\theta(\cdot \mid S_t)$ be the probability distribution of the action $a_t \sim \pi_\theta(\cdot \mid S_t)$ that will be taken given state S_t. Let $\xi = [S_0, a_0, S_1, a_1, ...]$ be a realization or a trajectory of the Markov process. To further improve the performance of the system, we can choose the parameter θ as:

$$\theta^* = \arg\max_{\theta\in\Theta} \mathbb{E}_{\xi\sim\pi_\theta} R(\xi) = \arg\max_{\theta\in\Theta} J(\theta), \tag{3.28}$$

where $R(\xi)$ is defined as the reward of the trajectory ξ, which can be any evaluation function of the system state such as the delay or the number of vehicle. $J(\theta)$ is the expected reward by taking the expectation with regard to the trajectory ξ: $J(\theta) = \mathbb{E}_{\xi\sim\pi_\theta} R(\xi)$. To optimize the parametric control policy $\theta \in \Theta$, which is the weight curve and the switching curve in this paper, we adopt the policy-gradient methods (Sutton and Barto, 2018) that update the parameter by using its gradient:

$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta J(\theta) \ _{\theta=\theta_k} \tag{3.29}$$

where α is the learning rate while the gradient is given by:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\xi\sim\pi_\theta}[\sum_t \nabla_\theta \log \pi_\theta(a_t|S_t).R(\xi)], \tag{3.30}$$

With the main idea of the policy gradient RL given by Equation (3.28-3.30), there have been different extensions based on it. For example, trust region policy optimization (TRPO) (Schulman et al., 2015) updates the policy by taking the largest step satisfying a special constraint on the distance between the new and old policies quantified by the KL divergence. To simplify the TRPO which solves a constrained optimization problem for each iteration, the proximal policy optimization (PPO) (Schulman et al., 2017) solves a proximal unconstrained optimization problem for each update. This paper will use the PPO to optimize the parameters in the max pressure policy network shown by Figure 3.7; more technical details will be discussed in Section 3.5.

3.5 Simulation Experiments

We use a simulation model built on SUMO (Krajzewicz et al., 2012) to compare the proposed max pressure control methods including ESCMP and LESCMP with the original max pressure control as the benchmark. Figure 3.8 shows the network used in the simulation, which is a corridor with six intersections on Plymouth Road, Ann Arbor, Michigan. The network topology is directly extracted from the OpenStreetMap data set (Haklay and Weber, 2008) while the traffic demand is calibrated from the historical data collected by videos during the evening peak hours. Figure 3.9 shows the traffic demand pattern used for the training of LESCMP. The duration of each episode is 60 min, which is divided into 5 different periods with variant demand levels or arrival rates. The vehicle arrival follows the Poisson process with a stationary arrival rate within each period. The relative demand level refers to the ratio of the realized demand to the calibrated peak-hour demand value. For the signal controller, each movement (controlled lane) will has an enforced 3-second yellow time when it is switched from green light to red light and a 2-second all-red clearance time.

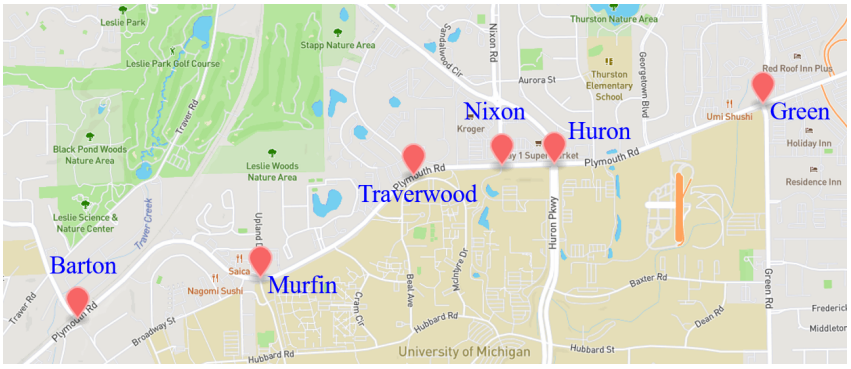


Figure 3.8: Network topology of Plymouth Rd., Ann Arbor, Michigan.

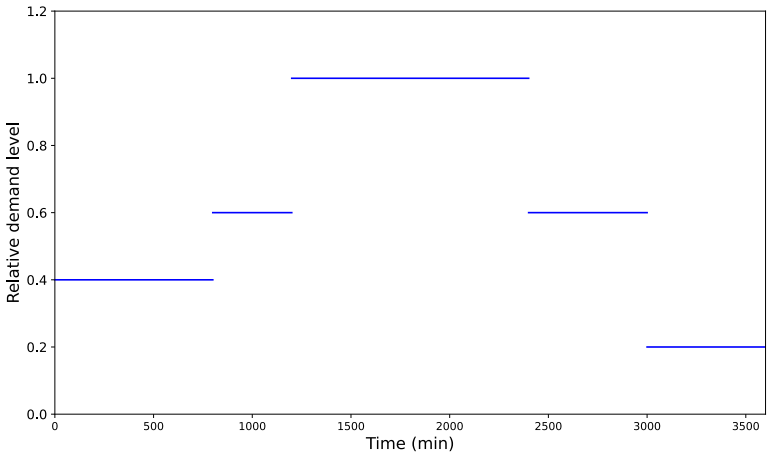


Figure 3.9: Input demand profile of the simulation environment for the training of the LESCMP.

There are totally three different controllers tested in the simulation study: PWBP, ESCMP, and LESCMP, where the position weighted back pressure control (PWBP) proposed by Li and Jabari (2019) is regarded as the benchmark. For the implementation of PWBP, we slightly change the position-based weight curve in the PWBP by using different curves for the running vehicles and the stopped vehicles as shown in Section 3.4.1. The weight curve for the running vehicles is chosen as the same as the PWBP while a uniform flat curve is used for the stopped vehicles. To avoid the unrealistic over-switching that will cause too much phase switching loss, we also set a 12-second minimum green for each phase for the implementation of PWBP. The proposed ESCMP tested in this section has the same weight curves with the PWBP but with an extra switching curve $F(x) = x^{0.4}$, where the parameter 0.4 is selected by a heuristic line search program.

Based on ESCMP, LESCMP is further optimized using the policy-gradient reinforcement learning algorithm (PPO). Essentially, ESCMP and LESCMP have the same controller structure given by Figure 3.7 while the parameters of LESCMP are well-tuned using RL algorithms with those of ESCMP as the initial point. For the implementation of the PPO that is used to train LESCMP, we use an open-source reinforcement learning library (i.e., Ray rllib, Liang et al., 2017) and Pytorch (Paszke et al., 2019) for automatic differentiation. The input state includes the traffic state as well as the current signal state. The real-time traffic state is represented by the number of the stopped and running vehicles within each cell of each lane while each lane is divided into cells for every 50 meters. The reward of the environment is chosen as the negative value of the total stop delay while a vehicle is regarded as stopped when its speed is less than a given threshold ($2m/s$). The policy network of the PPO is described in Section 3.4.1 while the switching curve is chosen as

$$F(x) = \alpha \cdot x^{\beta}, \tag{3.31}$$

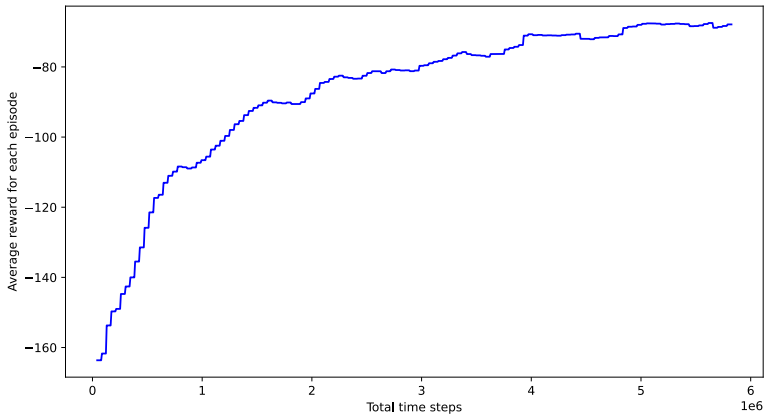


Figure 3.10: Training curve of the RL using the max pressure policy network.

where α and β are the parameters. Besides the policy network, the PPO also learns a value network at the same time, which is used as the critic to guide the update of the policy network. The value network is simply chosen as a two-layer fully-connected neural network with 256 neurons for each layer. The demand profile of the simulation is given by Figure 3.9 as aforementioned. Figure 3.10 shows the training curve of the PPO with the proposed max pressure policy network. The horizontal axis is the total time steps of the simulation environment while the vertical axis is the average scaled reward for each episode. As shown in the figure, the average reward increases along the training process, which means that the PPO keeps improving the LESCMP by adjusting the related parameters starting from the ESCMP.

We use the stop delay and the throughput to evaluate the traffic signal controllers. Since the temporal resolution of the simulation is 1 second, the stop delay for each time step is the total number of the stopped vehicles at the moment, which is also the total queue length of the network. By summing up the stop delay for each second, we can get the network total delay. The throughput refers to the vehicle that exit the network while the total throughput is the total number of vehicles that exit the network during the whole simulation horizon.

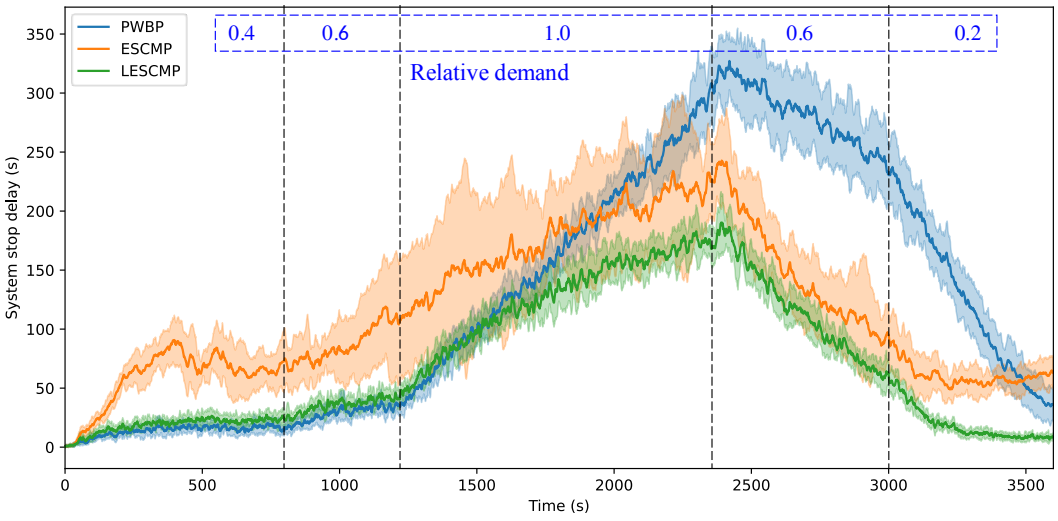


Figure 3.11: System stop delay under the different traffic signal controllers.

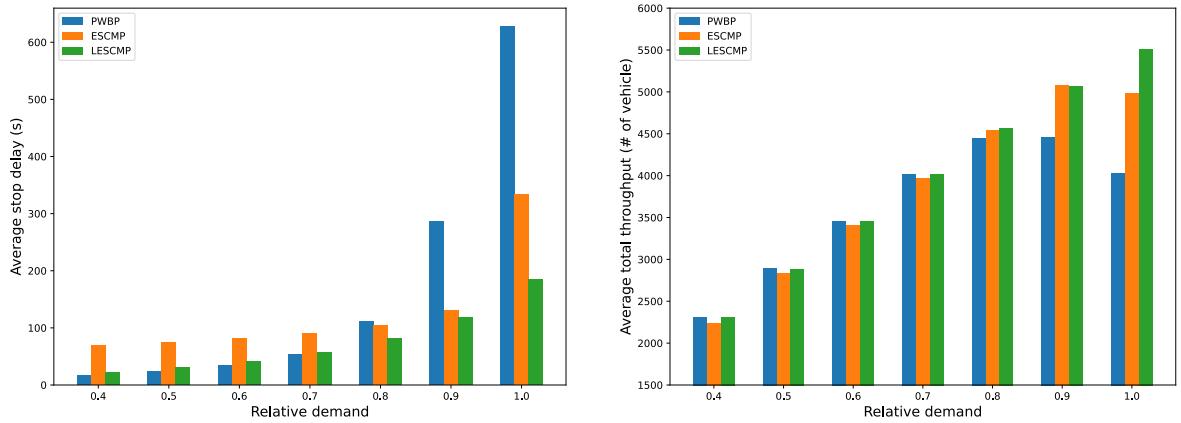
Under the demand profile given by Figure 3.9, Figure 3.11 and Table 3.1 show the comparison of the proposed max pressure methods with the benchmark controller. The evaluation for each controller is repeated for 10 times. For each evaluation, the input demand for all the controllers is generated

using the same random seed which means that the input demand is exactly the same for all the controllers. Figure 3.11 shows the mean and standard derivation of the system total delay of the three controllers while the Table 3.1 lists the average total delay and throughput associated with the standard deviation. As shown in Figure 3.11, under the low traffic demand, LESCMP and PWBP perform better than the ESCMP. In this situation, the switching loss will not influence the stability of the network queue lengths too much; and hence PWBP can perform well. However, when the traffic demand is high, PWBP cannot reduce the switching frequency accordingly; as a result, the network stability will be undermined, and queues will be built up quickly. Compared with the PWBP, ESCMP and LESCMP can dynamically adjust the phase switching frequency, so that they both have better performance than PWBP under higher traffic demand. Table 3.1 lists the mean and standard derivation of the system total delay and throughput of the three controllers. ESCMP performs slightly better than PWBP with regard to the total delay since it has better performance under the high traffic demand. As the well-tuned version of ESCMP, LESCMP further reduces the total delay by more than 35%. The total throughput of the three controllers is similar.

Table 3.1: Comparison of the system total delay of different controllers. Input demand is given by Figure 3.9.

Control policy	Total delay (h)	Delay std (h)	Total throughput (veh)	Throughput std (veh)
PWBP	129.15	8.8	3664.5	25.02
ESCMP	115.79	17.44	3659.9	10.74
LESCMP	72.15	2.63	3714.7	9.2

The evaluation results given by Figure 3.11 and Table 3.1 use the same input demand distribution with the training process of LESCMP, which might lead to an overestimation of the performance of LESCMP. Therefore, we design another "out-of-sample" experiment to test the three controllers under the different levels of stationary arrival rates. The evaluation is still repeated for 10 times for each controller under each demand. Figure 3.12 shows the average total delay and total throughput of the three controllers under different levels of traffic demand. Similar to the previous results, PWBP has the best low-demand performance but the worst high demand performance. The total delay of PWBP increases significantly when the traffic demand increases and the the total throughput starts to drop when the relative demand is higher than 0.8 due to the congestion. ESCMP has the worst low-demand performance but a much better high-demand performance than PWBP since the introduction of the switching curve can automatically reduce the switching frequency when the traffic volumes increase. Utilizing the same controller structure but with the further optimized parameters with regard to system delay, LESCMP performs well in all levels of traffic demand.



(a) Average total delay

(b) Average total throughput

Figure 3.12: Average total delay and throughput of the three controllers under different levels of (stationary) traffic demand.

3.6 Discussions and conclusions

This paper presents a noval framework that utilizes the policy-gradient reinforcement learning methods to learn a modified max pressure control policy considering the switching loss. The proposed max pressure control (SCMP) uses a switching rule that dynamically adjusts the switching frequency according to the congestion level. It is proved that SCMP is a throughput-optimal policy under the store-and-forward model with the phase switching loss. We also extend the theoretically derived SCMP to a more practical and flexible ESCMP with the weight curve and the switching curve. These two parametric curves are further optimized in LESCMP using the policy-gradient reinforcement learning algorithms. While the switching curve is used to address the switching loss caused by phase switching, the position weighted pressure can implicitly take the coordination between intersections into account. The simulation study established on a calibrated network showed that both ESCMP and LESCMP outperform the original max pressure control under the high traffic demand significantly and LESCMP performs well under all levels of traffic demand.

The proposed max pressure control methods shows many advantages in the real-world implementation. The control policy is decentralized so that each intersection makes its own decision based on the upstream and downstream traffic state without requiring communication between intersections. The switching rule enables the control policy to dynamically adjust the switching frequency (equivalent to cycle lengths) according to the congestion level so that we do not have to split the whole day into different time of days (TOD) and perform different signal timing plans or adjust the parameters. Besides, the max pressure control is an end-to-end control policy, which directly generates the control policy from the observation; hence it can be efficiently implemented in the real world.

This paper assumes that the real-time traffic density of each cell on every link is available. Therefore, real-time estimation of traffic density is required before implementing the proposed control method. There have been different methods proposed to estimate the real-time traffic state using different data resources (Nantes et al., 2016; Shahrbabaki et al., 2018).

From the reinforcement learning perspective, this paper mainly shows that our proposed max pressure control policy network can be used as the policy network in the policy-gradient reinforcement learning. The focus of this paper is the proposed policy network instead of the RL algorithms itself; and hence we use a baseline RL algorithm (PPO) and a centralized training method. There are certainly other more advanced RL algorithms and machine learning techniques that can be utilized such as multi-agent RL and transfer learning. Multi-agent RL can be naturally applied since the max pressure policy network is distributed among intersections. Besides, in this paper, we train different controllers for different intersections independently; essentially, it might be beneficial to design certain shared layer among different intersections since all the signal controllers have the same structure. We leave these more advanced RL algorithms based on the max pressure policy network for future study.

3.7 Appendix: table of notations

Notation	Meaning
Network model	
$\mathcal{G} = (\mathcal{N}, \mathcal{L})$	Traffic network $\mathcal{G}$ with the set of nodes $\mathcal{N}$ and the set of links $\mathcal{L}$
$i, j, k \in \mathcal{L}$	i, j, k are frequently-used indices for links
$\mathcal{M} = \mathcal{L} \times \mathcal{L}$	A movement is defined as a pair of links
$\mathcal{M}_o, \mathcal{M}_e$	The set of the ordinary/exit movements
$\mathcal{M}^n_{\text{in}}, \mathcal{M}^n_{\text{out}}$	The set of upstream/downstream movements of node n
x_{ij}, $\boldsymbol{x}$	Queue lengths of movement ij and the associated column vector
$s_{ij} \in \{0, 1\}, \boldsymbol{s}$	Signal state of movement ij and the associated column vector
$\mathcal{S}$	Feasible space of the signal state $\boldsymbol{s}$
$\lambda_{ij} \in \{0, 1\}$	Indicator whether the movement ij is in the discharge interval
$\boldsymbol{\Lambda}$	Diagonal matrix including all the λ_{ij}
T^r	Switching loss time
$a_{ij}, \boldsymbol{a}$	Exogenous arrival of movement ij and associated column vector
c_{ij}	Saturation flow rate of movement ij
$\boldsymbol{C}$	Diagonal matrix including all the c_{ij}
r_{ij}	Turning ratio from link i to link j
$\boldsymbol{R}$	Turning ratio matrix composed of r_{ij}
$S(t)$	Augmented system state at time t
SCMP controller and stability analysis	
$\psi(\cdot) : \mathbb{R} \to \mathbb{R}$	Switching function of SCMP determines the switching behavior
$\psi'(\cdot) : \mathbb{R} \to \mathbb{R}$	Switching function of the biased-based policy in Lemma 4
$F(\cdot) : \mathbb{R} \to \mathbb{R}$	A general monotonically increasing sublinear function
$\boldsymbol{w}(\cdot)$	The weight function of the queue lengths, $w_{ij}(\cdot) : \mathbb{R} \to \mathbb{R}, \forall ij \in \mathcal{M}$
$\text{pr}(\cdot, \cdot)$	Pressure function given network and traffic signal state
$\|\cdot\|$	1-norm of the column vector (summation of the absolute values)
$\mathcal{D}$	Admissible demand region of the $(\boldsymbol{a}, \boldsymbol{R})$ pair
$L(\cdot)$	Lyapunov function
Δ	Lyapunov drift
$\tilde{S}(t)$	Augmented system state at time t
τ_k	Time when the kth switching (change of the vector $\boldsymbol{s}$) is activated
Heuristic controller and RL	
$\tilde{w}(\cdot)$	The weight function given the vehicle locations along the movement, $\tilde{w}_{ij}(\cdot) : \mathbb{R}^n \to \mathbb{R}, \forall ij \in \mathcal{M}$
$w^*(\cdot)$	Position weighted curve, $w^*_{ij}(\cdot) : \mathbb{R} \to \mathbb{R}, \forall ij \in \mathcal{M}$
$l_{ij,p}$	Distance to the start of the movement ij of the pth vehicle in the movement
a_t	Action at the time t
S_t	System state at time t (the same as the augmented system state)
$\theta \in \Theta$	Parameters in the max pressure controller (the switching curve and the weight curve)
ξ	A realization or a trajectory of the system states and actions: $\xi = [S_0, a_0, S_1, a_1, ...]$
$R(\xi)$	Reward of a given trajectory ξ

Table 3.2: Notations and the corresponding meanings.

Chapter 4

An Implementable Traffic Signal Control Design using Reinforcement Learning Method

4.1 Introduction

Traffic signal control plays an important role in reducing travel time, mitigating traffic congestion, and decreasing greenhouse gas emissions. In the last few decades, the application of reinforcement learning (RL) in traffic signal control has attracted increasing attention due to its powerful learning ability and simple setting. Compared to the traditional optimization methods, it does not require a model of the system, and learns the policy directly by interacting with the environment.

Together with the convenience brought by using the RL methods, some research questions are also raised. Firstly, the design of state/reward/action structure plays a vital role in RL. (El-Tantawy and Abdulhai, 2010; Gao et al., 2017; Vidali et al., 2019; Zheng et al., 2019b) discuss the influences of state/reward design on single intersections, while for multiple intersections considered as individual agents, the situation becomes more complicated due to the non-stationary characteristic of the environment (Abdoos et al., 2011). Each agent is affected by other agents' policies, and it challenges not only the convergence but also the optimality of the algorithm. (Wei et al., 2019a; Lin et al., 2018) attempt to answer this question by investigating the trade-off between local reward of a single intersection and global reward of the traffic system. The second question is the coordination and scalability of the algorithms, especially for large networks, when centralized solutions are not feasible any more. As to the coordination, (Kuyer et al., 2008; El-Tantawy et al., 2013; Wei et al., 2019a; Balaji et al., 2010) develop different methodologies for the target intersection to communicate with other intersections. The graph attentional network ((Wei et al., 2019b)) and game theory such as the mean-field game ((Wang et al., 2020)) are utilized to deal with the scalability for large networks. Other directions such as vehicle connectivity ((Wiering, 2000; Xu et al., 2020; Huo et al., 2020)), disturbances ((Rasheed et al., 2020)), the geometry of the intersection ((Joo et al., 2020)), etc., are also explored in the literature.

While many aspects have been covered, there are still several problems need to be further discussed in the traffic signal control with RL. The first one is the lack of benchmarks. The results are typically compared with basic signal controllers such as fixed-time and Longest-queue-first (LQF) controllers (Thorpe, 1997; Abdulhai et al., 2003; El-Tantawy and Abdulhai, 2010; Abdoos et al., 2011; Gao et al., 2017), or controllers derived from other learning methods (Genders and Razavi, 2016; Wang et al., 2020; Li et al., 2016), or both (Joo et al., 2020; Rasheed et al., 2020; Xu et al., 2020). Some advanced controllers such as actuated controllers are used, but only in single intersections (Wang et al., 2019; Cabrejas-Egea et al., 2020; Mannion et al., 2015), or with simplified setting (Lin et al., 2018) (only the offset of the actuated controller is optimized and the action space is limited). A mixed control system including fixed-time controllers, semi-actuated controllers, and actuated controllers used by (El-Tantawy et al.,

2013) in a large network is one of the most realistic baselines, but it is not clear whether a direct comparison between the proposed RL controllers and the actuated controllers can still give the same level of improvement. (Balaji et al., 2010) uses GLIDE ((Keong, 1993), a modified version of SCATS) and two other multi-agent controllers as benchmarks, however, the algorithm requires online information sharing among agents, making it difficult to implement. Even though the learning performances show significant improvements, the conclusions are not convincing with those baseline controllers. The second problem is the implementation of the control algorithms. The assumption of global knowledge (such as the information from other agents (Balaji et al., 2010; Wang et al., 2020)), the limited choices of actions (Balaji et al., 2010; Wang et al., 2020; Gao et al., 2017; Li et al., 2016; Mannion et al., 2015), and the missing of yellow time (Abdoos et al., 2011) make the learning algorithms difficult to implement. Last but not the least, the accessibility of the data is rarely discussed. The data (e.g., waiting time or delay, travel time, velocity, and information from other intersections, etc.) used in training may not be easy to obtain in real life with the fixed location sensors. Detailed lists of data usage in different RL methods can be found in (Wei et al., 2019c).

In order to address these problems, this paper investigates an implementable deep multi-agent RL algorithm in a corridor level. The state of the RL agents can be obtained from the traffic cameras and loop detectors, and the action is designed with high flexibility based on the National Electrical Manufacturers Association (NEMA) dual-ring controllers, so that the algorithm is able to be implemented directly without changing the hardware of the current controllers. Note that although (Arel et al., 2010; Zang et al., 2020; Ma et al., 2021) consider the scheme on a single intersection environment, the NEMA convention is rarely followed in the design of the action space, especially in the multi-agent scheme. A safety constraint is constructed to guide the learning process and also to ensure vehicles from the non-arterial roads do not wait too long. The training episodes are generated from the ground truth collected from the peak traffic demand. The algorithm is tested in the Plymouth corridor from the City of Ann Arbor. The NEMA dual-ring actuated controllers which are implemented in the Plymouth corridor are built in SUMO (Simulation of Urban MObility) (Lopez et al., 2018) environment as the baseline, with the parameters optimized for the peak traffic demand. The simulation is conducted in SUMO, and the interaction between the observations and the traffic lights control is performed by TraCI (Traffic Control Interface).

The rest of the paper is organized as follows. A literature review is given in Sec. 2. Sec. 3 explains the methodology to be used and its settings. Sec. 4 introduces the Plymouth corridor from the City of Ann Arbor as the test environment, and establishes the actuated traffic signal controllers used in the corridor as the baseline. The training results and testing results are given and compared with the actuated controller in Sec. 5, followed by a conclusion in Sec. 6.

4.2 Literature Review

Based on the size of the transportation network, the study of RL in traffic signal control can be categorized into single-intersection and multi-intersection problems. Early research mainly applies the algorithms on single intersections (Thorpe, 1997; Abdulhai et al., 2003). With the development of the learning methods and computational power, more effort has been put on RL in multi-intersection setting (Wiering, 2000; Kuyer et al., 2008; El-Tantawy et al., 2013; Lin et al., 2018; Wei et al., 2019a; Chu et al., 2019; Wei et al., 2019b; Rasheed et al., 2020; Xu et al., 2020; Balaji et al., 2010). The challenges of multi-intersection problem lie on the coordination and scalability. When the size of the network increases, it seems to be more natural to used the distributed (or multi-agent) methods, since the large state space in the centralized fashion makes it computationally expensive. Much effort has been made to capture the cooperative-competitive relationship among agents, and balance the local-global optimum. (Kuyer et al., 2008) proposes a max-plus algorithm, it assumes the agents can broadcast the locally optimal information to the neighbours and optimize the joint reward with the neighbours. The contributions of the target agent and its neighbours are not distinguished, but the idea of achieving a global (sub-)optimal solution with local optimization is of great value. Similarly, (Balaji et al., 2010) coordinates by sharing traffic information (such as vehicle occupancy and count) of the outgoing links and the Q value with neighbouring agents in a network of 29 intersections. (Wei et al., 2019a) designs

the state and reward based on the theory of max pressure (Varaiya, 2013), which aims to maximize the throughput by minimizing the "pressure" (for example, the difference of vehicle numbers between incoming lanes and outgoing lanes). Some other studies use a weighted sum as the reward function: (Lin et al., 2018) forms the problem in a centralized fashion, and used a weighted sum of global reward and local reward as the reward function. (Chu et al., 2019) uses the advantage actor-critic (A2C) method, with the coordination obtained by using a weighted sum of queue lengths of the target intersection and the neighbouring intersections as the reward. (Huang et al., 2019) considers the outflow of the target intersection in the state representation to address the coordination issue. In terms of the scalability problem in large-scale network, (Wei et al., 2019b) uses the graph attentional network, distinguishes the contribution of different neighbouring agents, and creates an index-free model for agents to share their observations.(El-Tantawy et al., 2013) develops a controller with the tabular Q-learning method, the coordination is achieved by choosing the optimal action with the estimation the neighbours' behaviours (the probability of taking an action) at a joint state. The tabular methods suffer from the curse of dimensionality, that is, the size of the tables increases exponentially with the increase of number of agents. To address this problem, each agent plays a game with the neighbouring agents and avoids the direct interaction with them. A shared state representation and a mean-field game based reward structure are used in (Wang et al., 2020) to interact with other agents.

Recent development of technology on Dedicated Short-Range Communications (DSRC) and Cellular V2X (C-V2X) makes the vehicle connectivity particularly interesting in traffic signal control. In fact, as early as 2000, the idea of using connectivity in RL to improve the traffic signal control was proposed by (Wiering, 2000). It defines the value function on the vehicles, which enables a co-learning algorithm where the vehicles can also learn the path planning in the meanwhile. Different levels of communication are investigated, and the highest level requires the global knowledge of the system to compute the waiting times for all cars. Although 100% penetration rate of connected vehicles is still far from implementation, the vehicle trajectory data collected offline can still be advantageous. (Xu et al., 2020) utilizes the vehicle trajectory data to identify the critical nodes of a large-scale network, and develop a deep RL controller for each node independently while other nodes are controlled by fixed time controllers. Under the V2X communication assumption, (Huo et al., 2020) makes use of the trajectory data to pre-train the agents with imitation learning, and then fine-tunes the model with RL.

Traffic signal control on single intersections is of great importance to test the learning structures and understand the influence of a particular factor. Much work from recent years focuses on the state/reward design. (El-Tantawy and Abdulhai, 2010) discusses different state representations in the learning performance. (Gao et al., 2017) trains the convolutional neural network (CNN) with raw data (vehicle position and speed) instead of human-crafted data (such as queue length) being the state, and it extends the work of (Genders and Razavi, 2016) by considering a target network to improve the stability of the algorithm. (Vidali et al., 2019) investigates that the reward function can have a big impact on the training procedure. (Zheng et al., 2019b) discusses the state and reward design and interprets it with the transportation theory. With the state represented by the number of vehicles in the incoming lanes and the current traffic signal phase, and reward by queue length, the proposed design outperforms other state-action combinations under a binary action choice (to change the signal or not). By utilizing the similarity of different vehicle movements (left, through and right), (Zheng et al., 2019a) is able to capture the competitive relationship among different phases, and reduces the phase space, thus achieve fast convergence. (Joo et al., 2020) designs a state-action structure which can be extended to different geometries of intersections. Another related topic is the data used in state/reward representations. (Wang et al., 2019) represents the state by the green signal phase and the data purely extracted from the loop detectors. The training results show that the event-based data achieve a better performance than the aggregated data. (Cabrejas-Egea et al., 2020) discusses the different reward functions under accessible sensor data input. The average speed, queue length and stopped time based reward structures perform the best in various demands, and out of those the average speed based one has a smaller variation in waiting times. (Du et al., 2019) uses a trust region obtain the state, and defines a standardized reward function considering baseline queue lengths and average speed (corresponding to vehicle arrival rate).

Of course, the research on single intersections goes beyond the state/reward design. Both tech-

nology and methodology sides are investigated, such as the vehicle connectivity (Zhang et al., 2020), safety (Gong et al., 2020), emissions (Jie et al., 2021), and the use of deep neural network (NN) (Genders and Razavi, 2016; Li et al., 2016), etc. (Zhang et al., 2020) studies the influence of different penetration rate (or detection rate) of connected vehicles on the system performance. The simulation results suggest that even with a small portion of the connected vehicles, a shorter average waiting time can be achieved. (Gong et al., 2020) defines a risk score to evaluate the safety, and develops a multi-objective RL algorithm to incorporate the efficiency and safety concerns together. The efficiency and safety reward functions are calculated separately, and a weighted sum of the rewards is used to take actions. (Genders and Razavi, 2016) and (Li et al., 2016) utilize deep NNs, and compare the results with the controllers trained with shallow neural network (Genders and Razavi, 2016) or tabular method (Li et al., 2016). (Jie et al., 2021) considers vehicle emissions in the state and reward.

To summarize, RL on the multi-intersection problems mainly lies on the coordination and scalability, and the single-intersection problems tend to discuss more on the design of the learning algorithms. The vehicle connectivity is particularly interesting, and attracts increasing attention. Recently, some parallel learning and meta learning methods are proposed to increase learning speed by sharing and transferring knowledge. For example, (Mannion et al., 2015) uses a parallel learning method when several agents learn the same problem simultaneously with different instances, and they share the experience through an experience pool. The master agent can solve the problem on its own, while the slave agents influence the master agent by contributing to the experience pool. Meta learning aims at transferring knowledge learned from previous experiences so that the learning can be speeded up when it is applied to a new environment. In the work of (Zang et al., 2020), a value-based meta learning algorithm is introduced. The parameterized meta-learner is trained in different scenarios, and it is considered to be well-generalized with some initialized parameters. The performance is evaluated by applying the initialization to a new intersection.

In this study, we mainly focus on the coordination of the multi-intersection traffic signal control problem using a discounted shared reward. The NEMA phase scheme is kept in the action design and the local accessible information is used as state to ensure the implementation of the algorithm.

4.3 Methodology

Reinforcement learning (Sutton and Barto, 2018) refers to a type of machine learning methods that the agent learns to act by interacting with the environment iteratively. The traffic signal control problem is considered as an Markov Decision Process (MDP) (Sutton and Barto, 2018). The agent interacts with the environment at discrete time steps. At time step t, the agent observes the state s_t, and based on which an action a_t is taken. At the next time step $t+1$, a reward r_{t+1} is received. In an MDP, the state s_t and reward r_t only depend on the state s_{t-1} and action a_{t-1} of the previous time step, but not earlier. The probability $p(s_t, r_t|s_{t-1}, a_{t-1})$ describes the dynamics of the MDP.

A discount factor $\gamma \in [0, 1]$ is multiplied with the reward, and the goal of the agent is to maximize the cumulative discounted reward from all the steps. If $\gamma = 0$, the agent is myopic and only takes the immediate reward into account. If $\gamma = 1$, every step has the equal contribution to the cumulative result. The value of state s_t and action a_t is represented by $Q(s_t, a_t)$, and updated by:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \cdot \big(r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\big), \tag{4.1}$$

where α is the learning rate.

The problem is described as a multi-agent learning procedure. Each agent (traffic signal controller of one intersection) interacts with other agents, and the coordination is achieved by exchanging information. In the rest of this section, the state, action, and reward representations of the traffic signal control problem are designed, and after that, a deep neural network used by each agent is explained.

4.3.1 Design of System State, Reward, and Action

State

Considering the observation range of the traffic cameras, 150 m of the incoming lanes from the intersection is divided by 5 cells equally, and each cell is 30 m long. Regardless of the number of lanes corresponding to one through/right-turn or left-turn movement, the 30 m long distance is considered to be one cell. For example, the eastbound of a four-arm intersection in Fig. 4.1 has 10 cells, including 5 for the left-turn lane, and 5 for the through/right-turn lanes. The vector containing numbers of the vehicles in all the cells is used to represent the state.

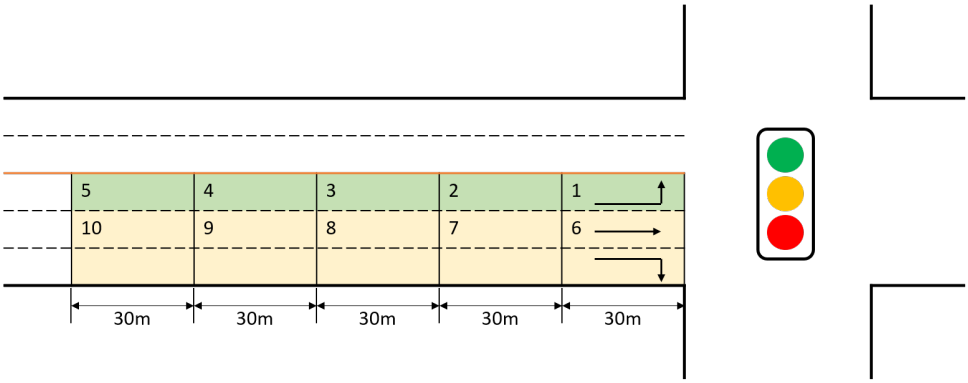


Figure 4.1: State representation for the eastbound of a four-arm intersection.

Only the traffic condition of the target intersection is observed in state, and the information can be obtained by the traffic surveillance cameras, which does not require extra knowledge from connectivity and is compatible with the current control arrangement.

Action

The action sets are designed based on the NEMA dual-ring structure of an actuated controller for a four-arm intersection. The agent can choose any non-conflict phase combination from the action sets (as shown in Fig. 4.2), and give them green time for the coming 3s. If the new action is the same with the current action, then the phase combination remains the same. If they are different, but on the same side of the barrier, the agent will switch to a transition action of 4s yellow time for the changing phase once the current action finishes, and then take the new action. If the new action is on the different side of the barrier, the agent will switch to transition actions of a 4s yellow time followed by a 2s all-red clearance time for both phases before taking the new action. If the current action is green time for phase 1 and phase 5, as shown in Fig. 4.3, different transition actions will be taken based on the next action. Note that no new action will be made until the next action has been executed, and this ensures the agent does not experience a strategy change during the transition action time.

Compared to the actuated controller, the action choices remain the same, and the 3s action time is the same with the extension time, but the flexibility is increased for the following reasons: a) the phase sequence is not fixed, and b) there is no minimum/maximum green time or cycle length. With this setting, the current action setting can be implemented to the corridor easily, and the performances of the different controllers become comparable.

Reward

The reward consists of three parts: the negative sum of queue lengths from all incoming lanes of the target intersection, a discounted sum of downstream lanes of the other intersections, and a waiting penalty.

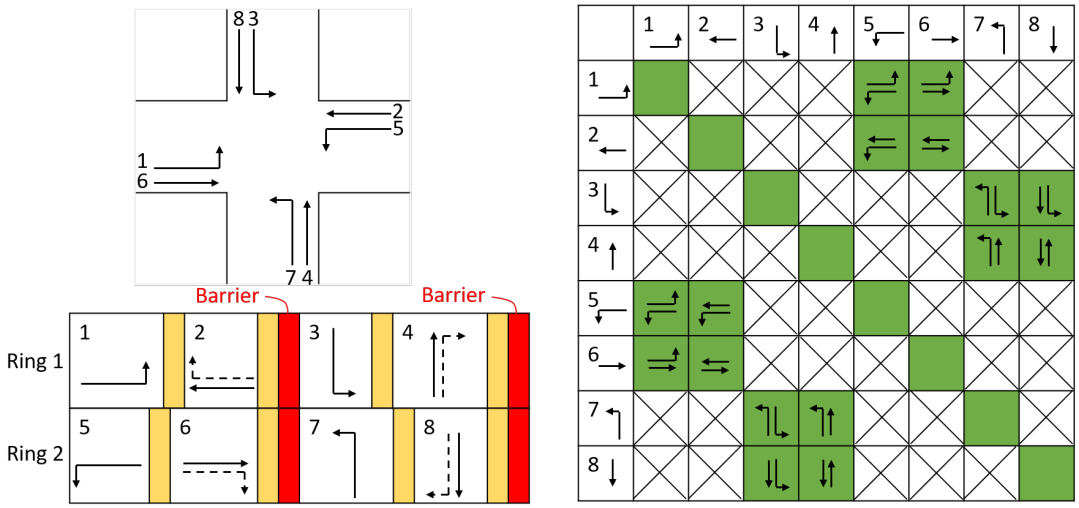


Figure 4.2: NEMA dual-ring phase combinations.

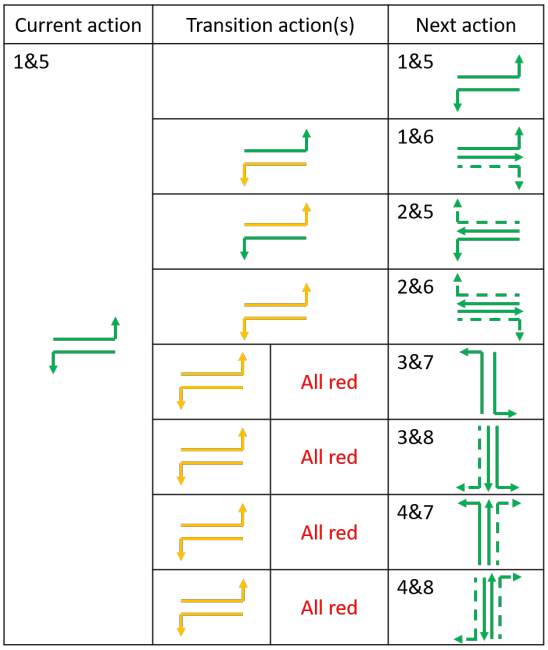


Figure 4.3: An example to illustrate the transition actions: if the current action is phase 1 and phase 5 green, the transition action(s) differs based on the next action. No new action will be made during the transition action time.

$$r = -q_i - \sum_{j \neq i, j \in N} \beta^d \cdot \tilde{q}_j - p_i, \tag{4.2}$$

where q_i is the sum of queue length of all the incoming lanes for agent i, $\tilde{q}_j$ is the sum of queue lengths for agent i's downstream lanes in all the other intersections $j \in N$ and N is the set of all intersections, β is the discount factor in space, $d \in \{1, 2, 3, ...\}$ is the distance between agent i and j, and p_i is the penalty for agent i if there is a vehicle standing in front of the traffic lights for more than 100s.

The reward structure is similar to that in (Chu et al., 2019), where the neighboring agents' rewards are discounted in space and added to the total reward. The discount factor is related to the spacial distance between two agents, the further the neighboring agent is, the smaller it contributes to the total reward. The item $\sum_{j \neq i, j \in N} \beta^d \cdot \tilde{q}_j$ in equation (4.2) keeps the above-mentioned property, but

differently, $\tilde{q}_j$ only takes into account the downstream queue lengths, since the only downstream is directly influenced by the action from agent i and is the most related component. In highly congested scenarios, when the queuing vehicles occupy the whole link and block other vehicles from crossing the intersection, the upstream queue lengths are also effected by the action of the target intersection.

The penalty p_i is set to guide the agent from the "unsafe" actions so that the training can converge faster. For the actuated controller, any phase is guaranteed with a minimum green time in every cycle, which ensures the traffic from the low demand roads can also pass the intersection. Similarly with (Gong et al., 2020), even without the concept of cycle length, this safety constraint can play such a role. If a vehicle waits in front of the traffic light for more than 100s, regardless of the "optimal" action by then, the agent will choose the next action which enables the corresponding phase to be green, and give a penalty to the reward function.

4.3.2 Deep Learning

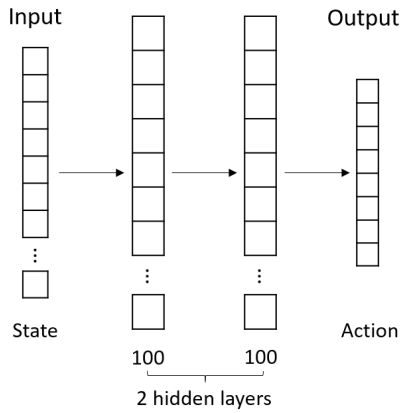


Figure 4.4: Neural network with two hidden layers.

A fully connected neural network (NN) with two hidden layers is constructed, and each layer has 100 neurons, see Fig. 4.4. The Rectified Linear Unit (ReLU) is used as the activation function. The number of input elements depends of the geometry of the intersection, specifically, the number of phases. For example, it is 40 for four-arm intersections, and 25 for three-arm intersections. The output is to set some non-conflict phases green, and the sizes of the action space differ in correspondence to the actuated controllers. The model is built with the help of Tensorflow (Abadi et al., 2015), using Adam as the optimizer.

4.4 Simulation Environment and Baseline Controller

The Plymouth corridor from the City of Ann Arbor, Michigan is built in SUMO. The length of the corridor is about 4100 meters, and the loop detectors are placed in front of the intersections (see the yellow rectangles in Fig. 4.5). There are four four-arm intersections including Murfin, Nixon, Huron, and Green, and two three-arm intersections including Barton and Traverwood.

A NEMA dual-ring actuated control scheme is used in the Plymouth corridor, and built in SUMO, with the parameters optimized for the peak traffic demand. It is called "actuated" since the control parameters are preset and actuated by the detectors, such as loop detectors. Through TraCI, the signals from the loop detectors are extracted, and used to set the traffic lights. There are three types of controllers based on the geometry and the traffic demand of the intersections. Murfin and Huron have all eight phases, Nixon and Green have six phases, and Barton and Traverwood have three phases, as shown in Fig. 4.6. Each phase has its minimum green time, and maximum green time. Once an active phase reaches the minimum green time, if the loop detectors detect the presence of more vehicles, the green time will be extended for a fixed amount of time (3s in this case) until the maximum green

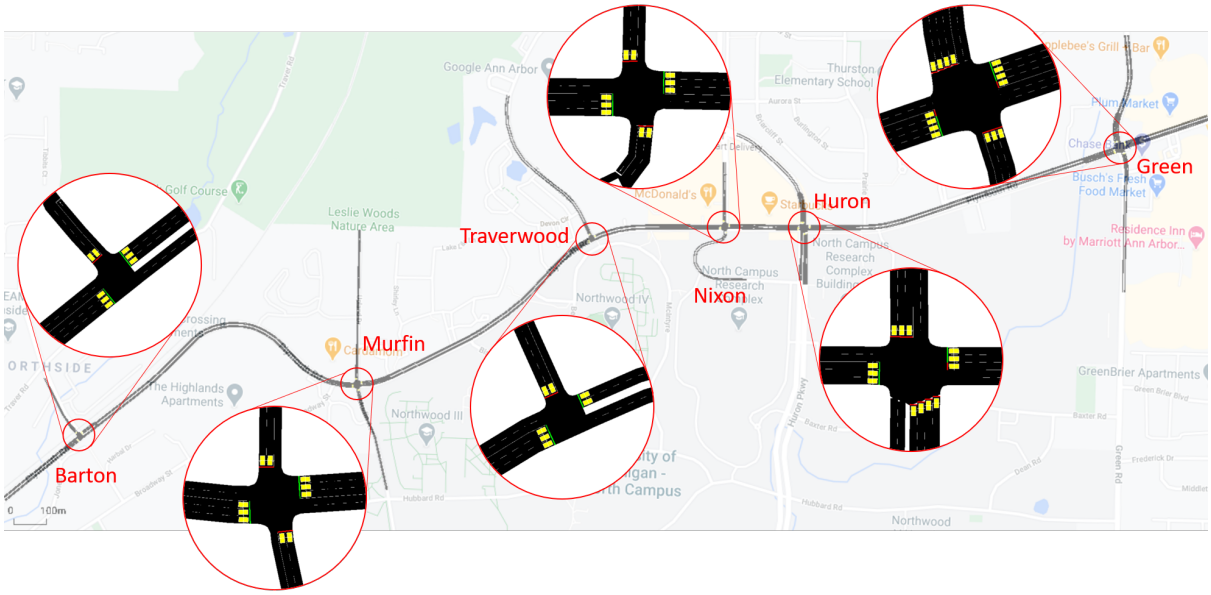


Figure 4.5: Plymouth corridor built in SUMO.

time. Each ring is actuated separately, and two rings are stopped together by a barrier. If a phase (for example, phase 2 of panel (a) in Fig. 4.6) of any ring reaches the maximum green time, both rings will be stopped regardless of the phase (in this case, phase 6) status of the other ring. If phase 2 has no vehicles waiting in the corresponding lanes before the maximum green time, and phase 6 is still actuated, then phase 2 will extend the green time until either of the phases reach the maximum green time. The six intersections are coordinated to ensure the green wave, with a cycle length of 100s, and the reference point starts from the end of phase 2. The yellow time is 4s for all the phases, and all-red clearance time is 2s.

Note that SUMO also has its built-in actuated traffic signal controller, but it does not follow the dual-ring structure, thus cannot be utilized in this study.

To accommodate the geometry and the actuated controllers of the intersections, the states and actions are adjusted accordingly. For Murfin, Nixon, Huron, and Green, there are 40 elements in the state vector, and for Barton and Traverwood, the number would be 25. The numbers of actions are 8, 6, and 2, for Murfin and Huron, Nixon and Green, and Barton and Traverwood, respectively. The action sets can be found in Table. 4.1.

Table 4.1: Action sets for different agents.

Intersections	Actions
Murfin, Huron	{1&5, 1&6, 2&5, 2&6, 3&7, 3&8, 4&7, 4&8}
Nixon, Green	{1&5, 1&6, 2&5, 2&6, 3, 4}
Barton, Traverwood	{2&6, 8}

4.5 Simulation Results

4.5.1 Training

The NN is trained for 200 episodes. During the simulation, all the samples are stored in a memory bank in the form of $(s_t, a_t, r_{t+1}, s_{t+1})$. When it reaches the memory capacity, the oldest samples will be deleted. Every simulation has 3600 seconds. The training is done with the experience replay after each simulation, for 400 epochs, with batch size of 64. The discount factor γ is 0.90, and the learning

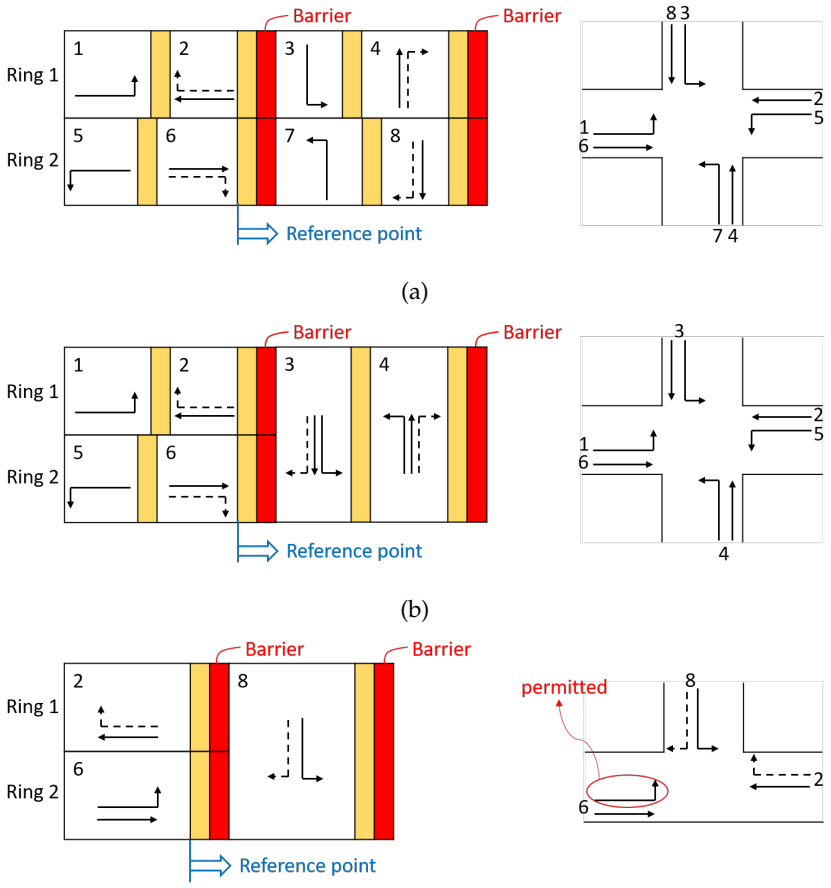


Figure 4.6: NEMA dual-ring structure of the actuated controllers in Plymouth corridor: (a) Murfin and Huron; (b) Nixon and Green; (c) Barton and Traverwood, where the left turn from eastbound is permitted.

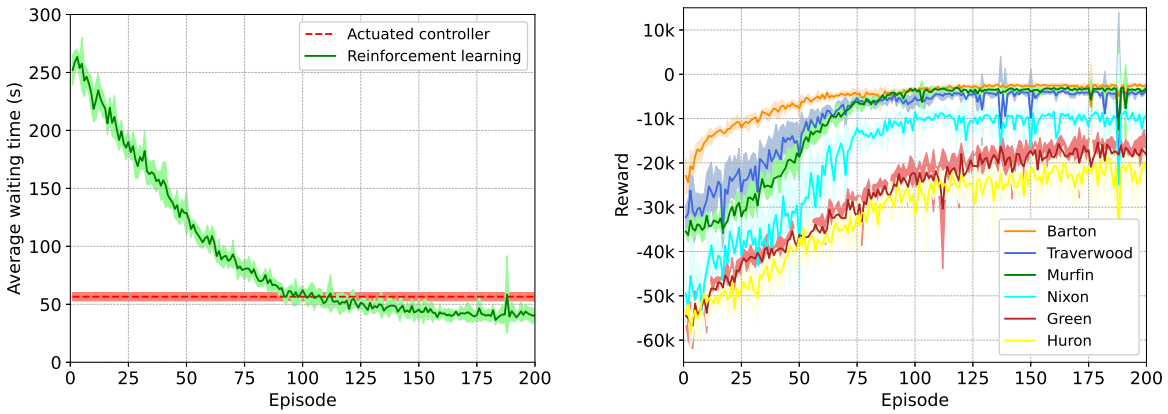
rate is set to 0.001. ϵ-greedy is used in the training to ensure exploration, meaning that with $\epsilon \in (0, 1)$ probability, the agent will choose a random action instead of the "best" action. The value is defined by:

$$\epsilon = \max\{0.01, 1 - \frac{t}{T}\} \tag{4.3}$$

where t is the number of episode, and T is the total number of episodes (200 in this case). ϵ decreases linearly from 1 to 0.01 in the first half episodes of the training, and keeps constant as 0.01 in the second half episodes. The pseudo code of the algorithm is shown in Algorithm 1.

A peak traffic demand collected from the Plymouth corridor is used to generate the route files in SUMO with different seeds. The actuated controllers run with same traffic demand are used as a baseline, and the average waiting time (or stop delay) caused by stops is plotted with mean and standard derivation in panel (a) of Fig. 4.7. The average waiting time is calculated by the total waiting time of vehicles arrived divided by number of vehicles arrived.

After about 110 episodes of training, the stop delay goes below that of the coordinated actuated controllers, see the green solid line in panel (a) of Fig. 4.7. The rewards of all six agents are plotted in panel (b) of Fig. 4.7. It is shown that the rewards for Barton, Traverwood, and Murfin converge to a higher level, followed by Nixon, and Green and Huron converge to the lowest values. That can be explained by the high demands in Green and Huron, which could lead to larger queue lengths. In addition to that, the distances with neighboring intersections are relatively long for Barton, Murfin, and even Traverwood, while Nixon is close to the busy intersection Huron, as depicted in Fig. 4.5.



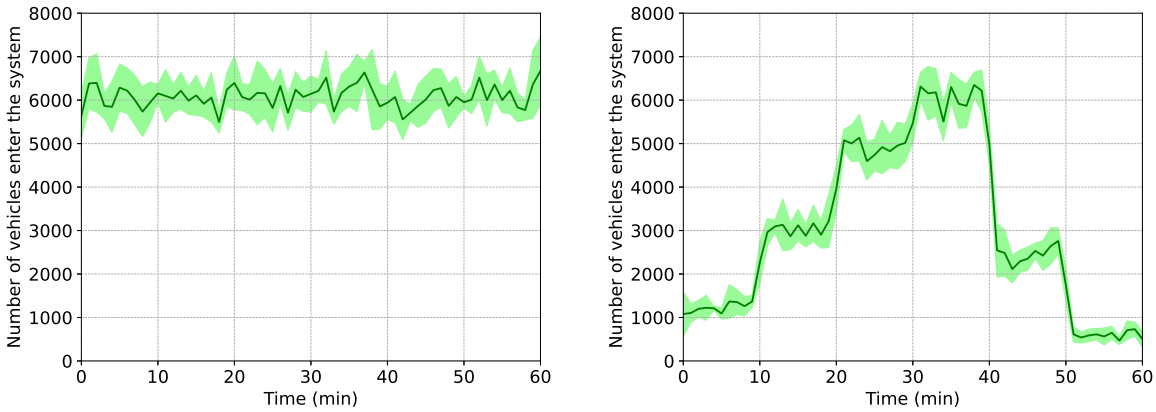
(a) Stop delay (delay caused by stops).

(b) Reward

Figure 4.7: Learning curves.

Algorithm 3: Deep Q-learning with safety constraint

episode $= 0$;
while episode < 200 **do**
  generate a route file for the simulation with a certain traffic demand;
  $\epsilon = \max(1 - \text{episode}/200,\ 0.01)$;
  initialize s_{t-1}, a_{t-1};
  while $t < 3600$ **do**
    for all the agents (intersections): **do**
      if the previous action a_t (after the transition action $\tilde{a}_t$) has finished: **then**
        get current state: s_t;
        get the waiting times for vehicles standing in front of the ego agent: t_k^w, for all the incoming lanes k;
        if $t_k^w == 100$ **then**
          take the action a_t which gives green to lane k;
          give a penalty p_i to the agent: $p_i = 100$;
        else
          $p_i = 0$.
        take action according to the exploration rate ϵ: $a_t(s_t, \epsilon)$;
        get the sum of queue lengths of all the incoming lanes for the ego agent: q_i;
        get the sum of queue lengths from the downstream lanes in the other intersections: $\tilde{q}_j (j \neq i, j \in N)$;
        calculate reward: $r_t = -q_i - \sum_{j \neq i, j \in N} \beta^d \cdot \tilde{q}_j - p_i$;
        add sample $(s_{t-1}, a_{t-1}, r_t, s_t)$ to the memory bank;
        $s_{t-1} \leftarrow s_t$, $a_{t-1} \leftarrow a_t$;
      else
        when $a_{t-1} \neq a_t$, take/execute the transition action $\tilde{a}_t$ (4s yellow, or 4s yellow + 2s all red) until it finishes, then take the new action a_t.
    $t \mathrel{+}= 1$;
  train the agents with experience replay, using samples from the memory banks;
  episode $\mathrel{+}= 1$



(a) Peak traffic demand

(b) Varying traffic demand

Figure 4.8: Traffic demands.

4.5.2 Testing

Two traffic demand patterns are created to validate the controller: a peak traffic demand and a varying traffic demand, see Fig. 4.8. The peak traffic demand in panel (a) has the same pattern with the training episodes but with different seeds, with the number of vehicles enter the corridor during the hour being about 6100. The varying demand in panel (b) shows the formation and dissipation of traffic, starting from about 1000 veh/h, reaching to the peak of about 6100 veh/h, and ending with about 700 veh/h. The same traffic demand patterns with the same seeds are tested with actuated controllers as baselines.

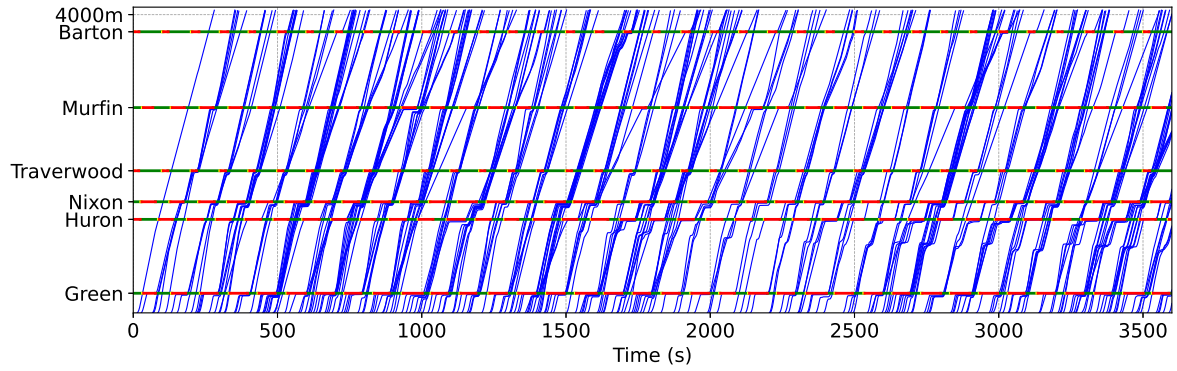
Table 4.2: Comparison of the performance measures between the actuated controller and the reinforcement learning controller.

		Actuated	RL	Improvement
Peak Traffic Demand	Stop Delay (s)	56.6 ± 3.5	39.0 ± 4.1	-31.1%
	Total Delay (s)	93.8 ± 4.2	76.9 ± 5.6	-18.0%
	Travel Time (s)	199.4 ± 4.7	182.7 ± 6.0	-8.3%
Varying Traffic Demand	Stop Delay (s)	43.8 ± 2.5	25.5 ± 2.4	-46.4%
	Total Delay (s)	74.4 ± 3.2	53.7 ± 3.1	-27.9%
	Travel Time (s)	182.5 ± 3.5	161.8 ± 3.5	-11.4%

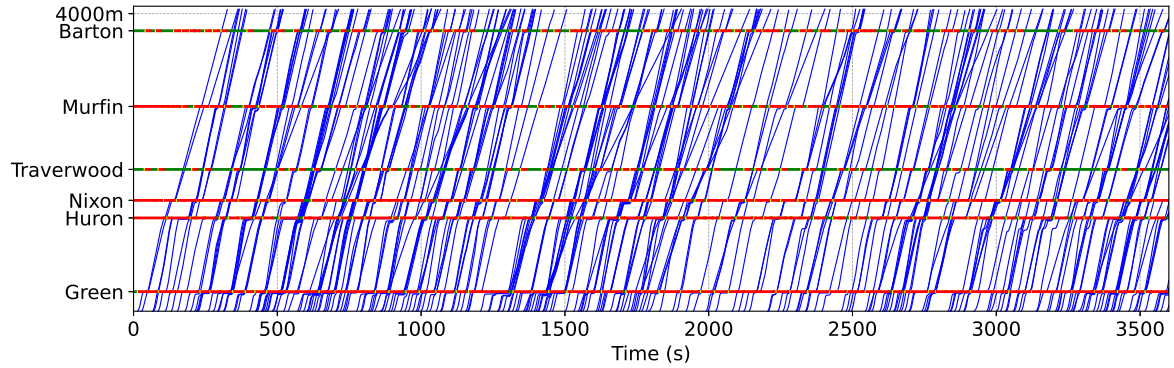
The comparison between the RL agents and the actuated controllers is shown in Tab. 4.2 with mean and standard derivation. Stop delay is the average delay caused by stops, total delay is the average delay caused by driving slower than free flow speed, and travel time is the average travel time from entering to leaving the corridor. All values are calculated based on vehicles which have left the corridor at the end of the simulation, and the running vehicles are not taken into account.

Under peak traffic demand, the mean value of stop delay is improved by 31.1% compared to the actuated controller, the total delay is reduced by 18.0%, and the travel time is decreased by 8.3%. For the varying traffic demand, the stop delay, total delay, and travel time are improved by 46.4%, 27.9%, and 11.4%, respectively.

The trajectories for vehicles running through the corridor from east to west under peak traffic demand are compared in Fig. 4.9. They are generated from the same route file, thus the vehicle departure times are the same. As shown in panel (b) of Fig. 4.7, there are three intersections: Green, Huron and Nixon, that are busier, and that is why the number of stops is higher in these intersections. In panel (a) of Fig. 4.9, vehicles are clustered by the red lights with actuated controllers, especially after Green intersection; while in panel (b) with RL agents, the trajectories are more distributed. The



(a)



(b)

Figure 4.9: Trajectories of vehicles running through the Plymouth corridor from east to west controlled by: (a) Actuated controllers, and (b) Reinforcement learning controllers.

traffic lights could accommodate the vehicles easily with the more flexible action settings in RL, and reduce the delay and travel time.

4.6 Conclusion

An implementable multi-agent reinforcement learning algorithm in the corridor level traffic signal control is proposed in this paper. Each intersection is an agent, and makes decision based on its own observations. The number of vehicles from 150 meters of the incoming lanes is used as the state, which can be extracted from the traffic surveillance cameras, and does not require the assumption of global information or information sharing (from connectivity). The coordination is achieved by the reward sharing structure. It consists of the sum of queue lengths from the target intersections, and a spatially discounted downstream queue lengths from other intersections, and a waiting penalty. The waiting penalty comes from the safety constraint, which gives a penalty when the vehicles are waiting in front of the traffic signals for more than 100 seconds. This can be obtained from the loop detectors, and ensures the vehicles from the minor roads do not have to wait too long.

Many learning methods suffer from the missing of a fair baseline. The RL controllers are typically compared with basic controllers (such as fixed time controller or longest-queue-first controller) or other learning methods. A NEMA dual-ring actuated control scheme from Plymouth corridor in city of Ann Arbor is built in SUMO as a benchmark. The action design is based on the dual-ring structure, any phase combinations from the dual-ring structure can be chosen as the next action. The agent gives a three-second green time to non-conflict phases, corresponding to the three-second extension time from the loop detectors. Compared to the actuated controllers, the RL action sets are more flexible since there is no fixed phase sequence, maximum/minimum green time, or cycle length. Furthermore,

it can be implemented to the current control scheme without modifying the firmware.

The peak traffic demand collected from the Plymouth corridor is used to generate simulations for training and testing. Compared to the benchmark, the proposed RL algorithm achieves a 31.1% reduction in stop delay under peak traffic demand, and the travel time is improved by 8.3%. Checking the trajectories of vehicles running through the corridor, the RL control has a more distributed trajectory pattern, which means that the vehicles can be released faster with shorter delay.

In a word, this paper addresses the data availability and implementation problem for traffic signal control with reinforcement learning, and establishes a benchmark based on the actuated controllers which are actually used in the Plymouth corridor.

The proposed algorithm assumes that the data from the cameras and loop detectors are flawless, and no V2I connectivity is available. Future work may focus on the influence of data uncertainty, and traffic signal control in the mixed traffic with connected automated vehicles (CAVs). The data from CAV trajectories can be utilized to further improve the performance, and the collaboration (or communication) between the CAVs and infrastructure will bring the problem to a new stage.

Chapter 5

Outcomes, Outputs and Impacts

Two journal papers have been formed from the project:
1. Wang, Xingmin, Yafeng Yin, Yiheng Feng, and Henry X. Liu. "Learning the max pressure control for urban traffic networks considering the phase switching loss." Transportation Research Part C: Emerging Technologies 140 (2022): 103670.
2. Xinyu Fei, Xingmin Wang, Xian Yu, Yiheng Feng, Henry Liu, Siqian Shen, Yafeng Yin, "Optimization and Decentralized Algorithms for Traffic Signal Control under Uncertain Traffic Demand and Vehicle Turning Ratio", under revision, 2021.

This project has the potential to improve the transportation network by optimizing traffic signal control, increasing throughput and reducing travel delay.

This report documents the findings from our preliminary investigations on real-time large-scale distributed control of traffic signals. Our investigations consist of three parts. Part 1 formulates a network traffic signal control problem while solving it via a distributed solution algorithm. Part 2 examines max pressure, a distributed control, and enhances it via reinforcement learning. In contrast, Part 3 directly applies multi-agent reinforcement learning to achieve distributed control.

More specifically, in Part 1, an mixed-integer program for the traffic signal control problem in urban traffic networks is built based on the cell transmission model. The deterministic model is extended to a two-stage stochastic model considering the uncertainties of traffic demand and turning ratios. Efficient algorithms are proposed for solving the models and overcoming the scalability difficulties. The proposed algorithm not only reduces the computational time but also ensures the optimality for the non-convex model with mixed-integer variables.

In Part 2, a noval framework that utilizes the policy-gradient reinforcement learning methods to learn a modified max pressure control policy considering the switching loss. The proposed max pressure control methods shows many advantages in the real-world implementation. The control policy is decentralized so that each intersection makes its own decision based on the upstream and downstream traffic state without requiring communication between intersections. The switching rule enables the control policy to dynamically adjust the switching frequency (equivalent to cycle lengths) according to the congestion level so that we do not have to split the whole day into different time of days (TOD) and perform different signal timing plans or adjust the parameters. Besides, the max pressure control is an end-to-end control policy, which directly generates the control policy from the observation; hence it can be efficiently implemented in the real world.

In Part 3, an implementable multi-agent reinforcement learning algorithm in traffic signal control is proposed. Each intersection is considered as an agent, and makes decision based on its own observations, while the coordination among agents are achieved by a shared reward structure. The data availability and implementation issues are addressed in the designing of state, action and reward. The safety is ensured by a fixed length of yellow time before taking the new action. Furthermore, a benchmark based on the actuated controllers is established, since many learning methods suffer from the missing of a fair baseline.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. URL: `https://www.tensorflow.org/`. software available from tensorflow.org.

Abdoos, M., Mozayani, N., Bazzan, A.L., 2011. Traffic light control in non-stationary environments based on multi agent q-learning, in: 2011 14th International IEEE conference on intelligent transportation systems (ITSC), IEEE. pp. 1580–1585.

Abdulhai, B., Pringle, R., Karakoulas, G.J., 2003. Reinforcement learning for true adaptive traffic signal control. Journal of Transportation Engineering 129, 278–285.

Aboudolas, K., Papageorgiou, M., Kosmatopoulos, E., 2009. Store-and-forward based methods for the signal control problem in large-scale congested urban road networks. Transportation Research Part C: Emerging Technologies 17, 163–174.

Adacher, L., Tiriolo, M., 2018. A macroscopic model with the advantages of microscopic model: A review of cell transmission model's extensions for urban traffic networks. Simulation Modelling Practice and Theory 86, 102–119.

Al Islam, S.B., Hajbabaie, A., 2017. Distributed coordinated signal timing optimization in connected transportation networks. Transportation Research Part C: Emerging Technologies 80, 272–285.

Arel, I., Liu, C., Urbanik, T., Kohls, A.G., 2010. Reinforcement learning-based multi-agent system for network traffic signal control. IET Intelligent Transport Systems 4, 128–135.

Balaji, P., German, X., Srinivasan, D., 2010. Urban traffic signal control using reinforcement learning agents. IET Intelligent Transport Systems 4, 177–188.

Boyd, S., Boyd, S.P., Vandenberghe, L., 2004. Convex optimization. Cambridge university press.

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al., 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends® in Machine Learning 3, 1–122.

Cabrejas-Egea, A., Howell, S., Knutins, M., Connaughton, C., 2020. Assessment of reward functions for reinforcement learning traffic signal control under real-world limitations. arXiv preprint arXiv:2008.11634 .

Celik, G., Borst, S.C., Whiting, P.A., Modiano, E., 2016. Dynamic scheduling with reconfiguration delays. Queueing Systems 83, 87–129.

Chen, R., Hu, J., Levin, M.W., Rey, D., 2020. Stability-based analysis of autonomous intersection management with pedestrians. Transportation Research Part C: Emerging Technologies 114, 463–483.

Chu, T., Wang, J., Codecà, L., Li, Z., 2019. Multi-agent deep reinforcement learning for large-scale traffic signal control. IEEE Transactions on Intelligent Transportation Systems 21, 1086–1095.

Daganzo, C., 1992. The cell transmission model. part i: A simple dynamic representation of highway traffic .

Du, Y., ShangGuan, W., Rong, D., Chai, L., 2019. Ra-tsc: Learning adaptive traffic signal control strategy via deep reinforcement learning, in: 2019 IEEE Intelligent Transportation Systems Conference (ITSC), IEEE. pp. 3275–3280.

El-Tantawy, S., Abdulhai, B., 2010. An agent-based learning towards decentralized and coordinated traffic signal control, in: 13th International IEEE Conference on Intelligent Transportation Systems, IEEE. pp. 665–670.

El-Tantawy, S., Abdulhai, B., Abdelgawad, H., 2013. Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (marlin-atsc): methodology and large-scale application on downtown toronto. IEEE Transactions on Intelligent Transportation Systems 14, 1140–1150.

Gao, J., Shen, Y., Liu, J., Ito, M., Shiratori, N., 2017. Adaptive traffic signal control: Deep reinforcement learning algorithm with experience replay and target network. arXiv preprint arXiv:1705.02755 .

Garcia Lopez, P., Montresor, A., Epema, D., Datta, A., Higashino, T., Iamnitchi, A., Barcellos, M., Felber, P., Riviere, E., 2015. Edge-centric computing: Vision and challenges.

Genders, W., Razavi, S., 2016. Using a deep reinforcement learning agent for traffic signal control. arXiv preprint arXiv:1611.01142 .

Gong, Y., Abdel-Aty, M., Yuan, J., Cai, Q., 2020. Multi-objective reinforcement learning approach for improving safety at intersections with adaptive traffic signal control. Accident Analysis & Prevention 144, 105655.

Haklay, M., Weber, P., 2008. Openstreetmap: User-generated street maps. IEEE Pervasive Computing 7, 12–18.

HCM, 2010. Transportation Research Board, National Research Council, Washington, DC 1207.

Hoogendoorn, S.P., Bovy, P.H., 2001. State-of-the-art of vehicular traffic flow modelling. Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering 215, 283–303.

Hsieh, P.C., Liu, X., Jiao, J., Hou, I.H., Zhang, Y., Kumar, P., 2017. Throughput-optimal scheduling for multi-hop networked transportation systems with switch-over delay, in: Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing, pp. 1–10.

Huang, R., Hu, J., Huo, Y., Pei, X., 2019. Cooperative multi-intersection traffic signal control based on deep reinforcement learning, in: CICTP 2019, pp. 2959–2970.

Huo, Y., Tao, Q., Hu, J., 2020. Cooperative control for multi-intersection traffic signal based on deep reinforcement learning and imitation learning. Ieee Access 8, 199573–199585.

Isa, N., Yusoff, M., Mohamed, A., 2014. A review on recent traffic congestion relief approaches, in: 2014 4th international conference on artificial intelligence with applications in engineering and technology, IEEE. pp. 121–126.

Jie, C., Wei, W., Zongli, L., Hong, Z., 2021. An intelligent traffic light control approach with vehicles emissions considered, in: 2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), IEEE. pp. 584–590.

Joo, H., Ahmed, S.H., Lim, Y., 2020. Traffic signal control for smart cities using reinforcement learning. Computer Communications 154, 324–330.

Keong, C.K., 1993. The glide system—singapore's urban traffic control system. Transport reviews 13, 295–305.

Khamis, M.A., Gomaa, W., 2014. Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework. Engineering Applications of Artificial Intelligence 29, 134–151.

Koonce, P., Rodegerdts, L., 2008. Traffic signal timing manual. Technical Report. United States. Federal Highway Administration.

Krajzewicz, D., Erdmann, J., Behrisch, M., Bieker, L., 2012. Recent development and applications of sumo-simulation of urban mobility. International journal on advances in systems and measurements 5.

Kuyer, L., Whiteson, S., Bakker, B., Vlassis, N., 2008. Multiagent reinforcement learning for urban traffic control using coordination graphs, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer. pp. 656–671.

Le, T., Kovács, P., Walton, N., Vu, H.L., Andrew, L.L., Hoogendoorn, S.S., 2015. Decentralized signal control for urban road networks. Transportation Research Part C: Emerging Technologies 58, 431–450.

Levin, T., 2020. The 31 US cities that had the worst traffic in 2019 according to a study. `https://www.businessinsider.com/us-cities-most-traffic-2019-2020-3`.

Li, K., Ioannou, P., 2004. Modeling of traffic flow of automated vehicles. IEEE Transactions on Intelligent Transportation Systems 5, 99–113.

Li, L., Jabari, S.E., 2019. Position weighted backpressure intersection control for urban networks. Transportation Research Part B: Methodological 128, 435–461.

Li, L., Lv, Y., Wang, F.Y., 2016. Traffic signal timing via deep reinforcement learning. IEEE/CAA Journal of Automatica Sinica 3, 247–254.

Liang, E., Liaw, R., Nishihara, R., Moritz, P., Fox, R., Gonzalez, J., Goldberg, K., Stoica, I., 2017. Ray rllib: A composable and scalable reinforcement learning library. arXiv preprint arXiv:1712.09381 , 85.

Liberzon, D., 2003. Switching in systems and control. Springer Science & Business Media.

Lin, Y., Dai, X., Li, L., Wang, F.Y., 2018. An efficient deep reinforcement learning model for urban traffic control. arXiv preprint arXiv:1808.01876 .

Lioris, J., Kurzhanskiy, A.A., Triantafyllos, D., Varaiya, P., 2014. Control experiments for a network of signalized intersections using the '. q'simulator. IFAC Proceedings Volumes 47, 332–337.

Little, J.D., Graves, S.C., 2008. Little's law, in: Building intuition. Springer, pp. 81–100.

Lo, H.K., 1999. A novel traffic signal control formulation. Transportation Research Part A: Policy and Practice 33, 433–448.

Lopez, P.A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., Wießner, E., 2018. Microscopic traffic simulation using sumo, in: 2018 21st International Conference on Intelligent Transportation Systems (ITSC), IEEE. pp. 2575–2582.

Ma, D., Zhou, B., Song, X., Dai, H., 2021. A deep reinforcement learning approach to traffic signal control with temporal traffic pattern mining. IEEE Transactions on Intelligent Transportation Systems .

Mannion, P., Duggan, J., Howley, E., 2015. Parallel reinforcement learning for traffic signal control. Procedia Computer Science 52, 956–961.

Manolis, D., Pappa, T., Diakaki, C., Papamichail, I., Papageorgiou, M., 2018. Centralised versus decentralised signal control of large-scale urban road networks in real time: a simulation study. IET Intelligent Transport Systems 12, 891–900.

Nantes, A., Ngoduy, D., Bhaskar, A., Miska, M., Chung, E., 2016. Real-time traffic state estimation in urban corridors from heterogeneous data. Transportation Research Part C: Emerging Technologies 66, 99–118.

Neely, M.J., 2010. Stochastic network optimization with application to communication and queueing systems. Synthesis Lectures on Communication Networks 3, 1–211.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al., 2019. Pytorch: An imperative style, high-performance deep learning library. arXiv preprint arXiv:1912.01703 .

Pipes, L.A., 1953. An operational analysis of traffic dynamics. Journal of applied physics 24, 274–281.

Rasheed, F., Yau, K.L.A., Low, Y.C., 2020. Deep reinforcement learning for traffic signal control under disturbances: A case study on sunway city, malaysia. Future Generation Computer Systems 109, 431–445.

Schrank, D., Eisele, B., T., L., 2019. 2019 Urban mobility report. Technical Report. Texas A&M Transportation Institute.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., Moritz, P., 2015. Trust region policy optimization, in: International conference on machine learning, pp. 1889–1897.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O., 2017. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 .

Shahrbabaki, M.R., Safavi, A.A., Papageorgiou, M., Papamichail, I., 2018. A data fusion approach for real-time traffic state estimation in urban signalized links. Transportation research part C: emerging technologies 92, 525–548.

Shapiro, A., 1993. Asymptotic behavior of optimal solutions in stochastic programming. Mathematics of Operations Research 18, 829–845.

Shapiro, A., Dentcheva, D., Ruszczyński, A., 2014. Lectures on stochastic programming: modeling and theory. SIAM.

Shapiro, A., Homem-de Mello, T., 2000. On the rate of convergence of optimal solutions of monte carlo approximations of stochastic programs. SIAM Journal on Optimization 11, 70–86.

Srikant, R., Ying, L., 2013. Communication networks: an optimization, control, and stochastic networks perspective. Cambridge University Press.

Sun, X., Yin, Y., 2018. A simulation study on max pressure control of signalized intersections. Transportation research record 2672, 117–127.

Sutton, R.S., Barto, A.G., 2018. Reinforcement learning: An introduction. MIT press.

Tang, K., Boltze, M., Nakamura, H., Tian, Z., 2019. Global Practices on Road Traffic Signal Control: Fixed-time Control at Isolated Intersections. Elsevier.

Tassiulas, L., Ephremides, A., 1990. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks, in: 29th IEEE Conference on Decision and Control, IEEE. pp. 2130–2132.

Thorpe, T.L., 1997. Vehicle traffic light control using sarsa, in: Online]. Available: citeseer. ist. psu. edu/thorpe97vehicle. html, Citeseer.

Timotheou, S., Panayiotou, C.G., Polycarpou, M.M., 2014. Distributed traffic signal control using the cell transmission model via the alternating direction method of multipliers. IEEE Transactions on Intelligent Transportation Systems 16, 919–933.

Tong, Y., Zhao, L., Li, L., Zhang, Y., 2015. Stochastic programming model for oversaturated intersection signal timing. Transportation Research Part C: Emerging Technologies 58, 474–486.

Varaiya, P., 2013. Max pressure control of a network of signalized intersections. Transportation Research Part C: Emerging Technologies 36, 177–195.

Vidali, A., Crociani, L., Vizzari, G., Bandini, S., 2019. A deep reinforcement learning approach to adaptive traffic lights management., in: WOA, pp. 42–50.

Wang, S., Xie, X., Huang, K., Zeng, J., Cai, Z., 2019. Deep reinforcement learning-based traffic signal control using high-resolution event-based data. Entropy 21, 744.

Wang, X., Ke, L., Qiao, Z., Chai, X., 2020. Large-scale traffic signal control using a novel multiagent reinforcement learning. IEEE transactions on cybernetics 51, 174–187.

Wang, X., Yin, Y., Feng, Y., Liu, H.X., 2022. Learning the max pressure control for urban traffic networks considering the phase switching loss. Transportation Research Part C: Emerging Technologies 140, 103670.

Webster, F.V., 1958. Traffic signal settings. Technical Report.

Wei, H., Chen, C., Zheng, G., Wu, K., Gayah, V., Xu, K., Li, Z., 2019a. Presslight: Learning max pressure control to coordinate traffic signals in arterial network, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1290–1298.

Wei, H., Xu, N., Zhang, H., Zheng, G., Zang, X., Chen, C., Zhang, W., Zhu, Y., Xu, K., Li, Z., 2019b. Colight: Learning network-level cooperation for traffic signal control, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 1913–1922.

Wei, H., Zheng, G., Gayah, V., Li, Z., 2019c. A survey on traffic signal control methods. arXiv preprint arXiv:1904.08117 .

Wiering, M.A., 2000. Multi-agent reinforcement learning for traffic light control, in: Machine Learning: Proceedings of the Seventeenth International Conference (ICML'2000), pp. 1151–1158.

Wu, N., Brilon, W., 1999. Cellular automata for highway traffic flow simulation. Proceedings 14th International Symposium on Transportation and Traffic Theory (Abbreviated presentations) .

Xiao, N., Frazzoli, E., Li, Y., Wang, Y., Wang, D., 2014. Pressure releasing policy in traffic signal control with finite queue capacities, in: 53rd IEEE Conference on Decision and Control, IEEE. pp. 6492–6497.

Xu, M., Wu, J., Huang, L., Zhou, R., Wang, T., Hu, D., 2020. Network-wide traffic signal control based on the discovery of critical nodes and deep reinforcement learning. Journal of Intelligent Transportation Systems 24, 1–10.

Yau, K.L.A., Qadir, J., Khoo, H.L., Ling, M.H., Komisarczuk, P., 2017. A survey on reinforcement learning models and algorithms for traffic signal control. ACM Computing Surveys (CSUR) 50, 1–38.

Yu, X.H., Recker, W.W., 2006. Stochastic adaptive control model for traffic signal systems. Transportation Research Part C: Emerging Technologies 14, 263–282.

Yuan, Y., Wang, F.Y., 2016. Towards blockchain-based intelligent transportation systems, in: 2016 IEEE 19th international conference on intelligent transportation systems (ITSC), IEEE. pp. 2663–2668.

Zaidi, A.A., Kulcsár, B., Wymeersch, H., 2016. Back-pressure traffic signal control with fixed and adaptive routing for urban vehicular networks. IEEE Transactions on Intelligent Transportation Systems 17, 2134–2143.

Zang, X., Yao, H., Zheng, G., Xu, N., Xu, K., Li, Z., 2020. Metalight: Value-based meta-reinforcement learning for traffic signal control, in: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 1153–1160.

Zhang, G., Wang, Y., 2010. Optimizing minimum and maximum green time settings for traffic actuated control at isolated intersections. IEEE Transactions on Intelligent Transportation Systems 12, 164–173.

Zhang, L., Yin, Y., Lou, Y., 2010. Robust signal timing for arterials under day-to-day demand variations. Transportation Research Record 2192, 156–166.

Zhang, R., Ishikawa, A., Wang, W., Striner, B., Tonguz, O.K., 2020. Using reinforcement learning with partial vehicle detection for intelligent traffic signal control. IEEE Transactions on Intelligent Transportation Systems .

Zheng, G., Xiong, Y., Zang, X., Feng, J., Wei, H., Zhang, H., Li, Y., Xu, K., Li, Z., 2019a. Learning phase competition for traffic signal control, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 1963–1972.

Zheng, G., Zang, X., Xu, N., Wei, H., Yu, Z., Gayah, V., Xu, K., Li, Z., 2019b. Diagnosing reinforcement learning for traffic signal control. arXiv preprint arXiv:1905.04716 .